The GSAM Final Report has been divided into three volumes. These consist of:

Volume 1:		Summary Report
Volume II:		GSAM User's Guide
Volume III:		GSAM Programmer Guide
	IIIa:	Programmer's Guide for the Reservoir Performance Module of GSAM
	IIIb:	Programmer's Guide for the Storage Reservoir Performance Module of GSAM
	IIIIc:	Programmer's Guide for the Exploration and Production Module of GSAM
	IIId:	Programmer's Guide for the Demand and Integrating Module of GSAM

ICF
CONSULTING

DEVELOPMENT OF A NATURAL GAS SYSTEMS ANALYSIS MODEL (GSAM)

FINAL REPORT

Volume I – Summary Report

For:

U.S. Department of Energy
National Energy Technology Laboratory
Morgantown, West Virginia
Under Contract Number: DE-AC21-92MC28138

By:

ICF Consulting, Inc.
Fairfax, Virginia

February 2001

ICF CONSULTING

DISCLAIMER

TABLE OF CONTENTS

TABLE OF CONTENTS
(CONTINUED)

LIST OF FIGURES/TABLES

EXECUTIVE SUMMARY

This report summarizes work completed on DOE Contract DE-AC21-92MC28138, Development of a Natural Gas Systems Analysis Model (GSAM). The products developed under this project directly support the National Energy Technology Laboratory (NETL) in carrying out its natural gas R&D mission.

The objective of this research effort has been to create a comprehensive, non-proprietary, microcomputer model of the North American natural gas market. GSAM has been developed to explicitly evaluate components of the natural gas system, including the entire in-place gas resource base, exploration and development technologies, extraction technology and performance parameters, transportation and storage factors, and end-use demand issues. The system has been fully tested and calibrated and has been used for multiple natural gas metrics analyses at NETL in which metric associated with NETL natural gas upstream R&D technologies and strategies under the direction of NETL has been evaluated.

NETL's Natural Gas Strategic Plan requires that R&D activities be evaluated for their ability to provide adequate supplies of reasonably priced natural gas. GSAM provides the capability to assess potential and on-going R&D projects using a full fuel cycle, cost-benefit approach. This method yields realistic, market-based assessments of benefits and costs of alternative or related technology advances. GSAM is capable of estimating both technical and commercial successes, quantifying the potential benefits to the market, as well as to other related research. GSAM, therefore, represents an integration of research activities and a method for planning and prioritizing efforts to maximize benefits and minimize costs. Without an analytical tool like GSAM, NETL natural gas upstream R&D activities cannot be appropriately ranked or focused on the most important aspects of natural gas extraction efforts or utilization considerations.

The final documentation of this research effort is divided in three-volume report as follows:

Volume I, *Summary Report*, provides overall summary of the GSAM research effort.

Volume II, *User's Guide*, provides details on the data, models, required inputs, input and output formats, and nuts and bolts for successfully running GSAM. The primary purpose of this volume is to provide information necessary to configure and operate the model for many analytical purposes.

Volume III, *Programmer's Guides*, provides detailed description of the computer code for various GSAM modules.

Volume IIIa contains programmer’s guide for the Reservoir Performance Module of GSAM.

Volume IIIb contains programmer’s guide for the Storage Reservoir Performance Module of GSAM.

Volume IIIc contains programmer’s guide for the Exploration and Production Module of GSAM.

Volume IIId contains programmer’s guide for the Demand and Integrating Module of GSAM.

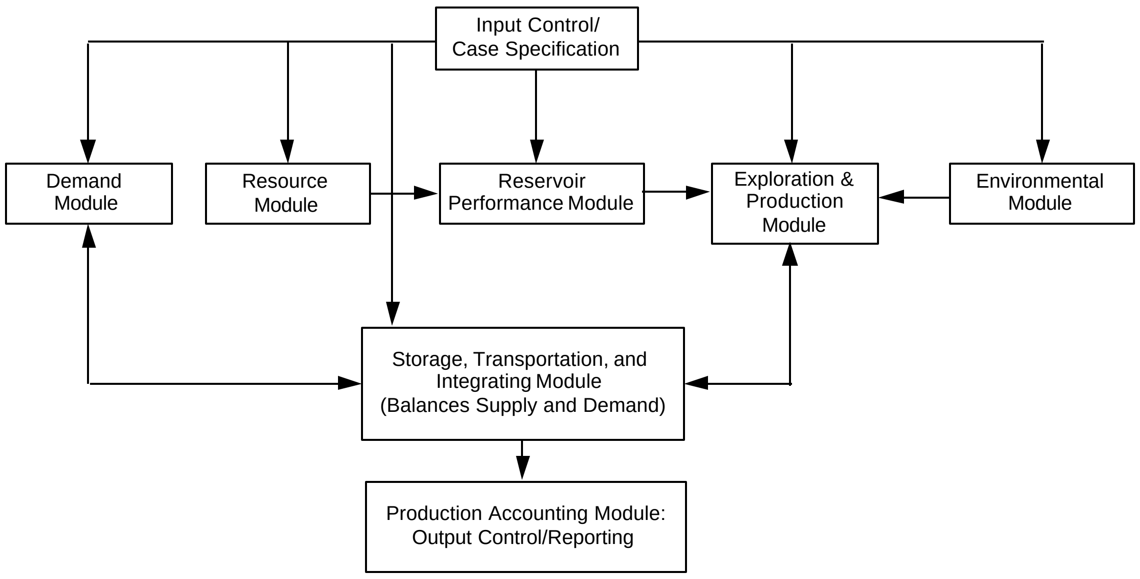
I. OVERVIEW OF GSAM

I.1 Introduction

The Gas Systems Analysis Model (GSAM) has been developed for National Energy Technology Laboratory (NETL), Department of Energy (DOE), to simultaneously evaluate the impact of various combinations of technologies and energy policies on the overall North American natural gas industry. This gas market is an integrated, commodity-based system of supply, transportation, and demand, which recent regulatory changes have dramatically altered. Demand and supply of gas are influenced and limited by market conditions and regional boundaries established by the existing infrastructure.

The consistent evaluation of gas supply and demand under alternative economic, technology, regulatory, and policy conditions is the main objective of the modeling system, which is designed to be fully consistent with operator decision-making procedures. Its modular design provides flexibility in developing and completing detailed resource assessments. Figure 1.1 shows the general modular structure and logic flow for the entire system.

Figure 1.1
Structure of the Gas Systems Analysis Model



Input Control/ Case Specification
Demand Module
Resource Module
Reservoir Performance Module
Exploration & Production Module
Environmental Module
Storage, Transportation, and Integrating Module (Balances Supply and Demand)
Production Accounting Module: Output Control/Reporting

I.2 System Description

The Gas Systems Analysis System has been designed to fully assess the benefits and costs associated with the flow of gas from the reservoir to various end-users. It can be used to evaluate the potential of alternative R&D strategies to increase natural gas extraction efficiency and/or reduce costs. Many existing models are already designed to evaluate various aspects of natural gas recovery at the level of an individual well. The uniqueness of the system lies in its simultaneous evaluation of R&D strategy in a market context, with the benefits of technology advances measured in terms of both commercial and technical success.

Required Inputs

Reservoir Data. Since model results are only as useful as the data on which they are based, the methodology places a high priority on using validated information on natural gas resources in the United States and Canada. The methodology relies on accurate and available resource descriptions, at the level of individual reservoirs.

Each reservoir is explicitly characterized by rock and fluid properties, depth, pressure, and temperature, as well as resource type, play, location, and current development status. These data, plus carefully selected defaults when required, are used to evaluate the productivity of each reservoir for a set of alternative technology & policy scenarios.

By being able to evaluate known reservoirs, GSAM's comprehensive characterization of the resource provides a credible basis for a variety of analyses.
GSAM can assess the impacts of various technology and economic scenarios on the reserves. The objective is to characterize and evaluate exploration and production options as an operator would in the field. After examination of several publicly available databases, NRG Associates' (NRG) Significant Oil and Gas Fields Database was determined to be the best reservoir database available to accomplish this goal.

The NRG database is a reservoir-level database for both the known and undeveloped natural gas resources and has sufficient coverage of necessary data elements to do a volumetric calculation of OGIP. It also contains sufficient data to perform analyses on different resource types, different regions, and other levels of disaggregation.

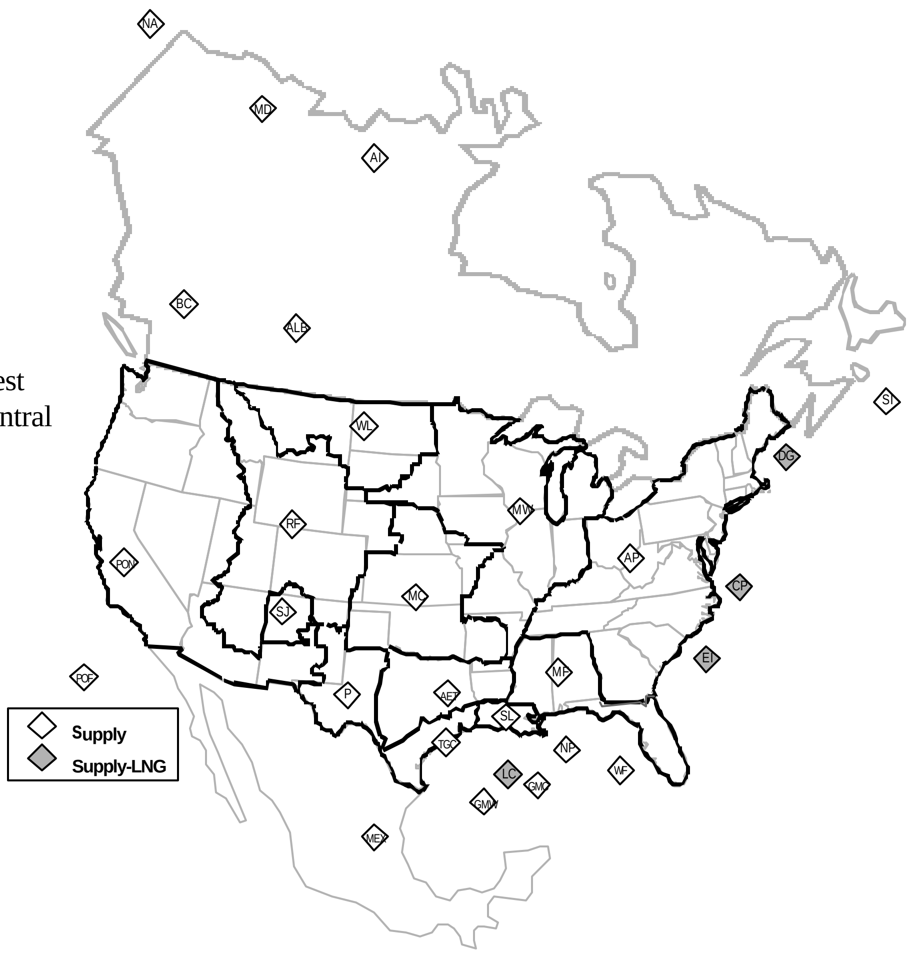
The NRG database contains detailed information on over 17,000 natural gas reservoirs in the U.S. and Canada, accounting for about 90% of the total North American gas potential from known fields (excluding Appalachia). Nearly 85% of the reservoirs in the NRG database have information on production history and proved reserves.

Each of the reservoirs has detailed information on the location of the reservoir, allowing the allocation of reservoirs into the regions (Figure 1.2). Producing regions are allocated in the system according to their geologic characteristics. Each reservoir is also assigned to a play. Plays are clusters of geologically similar reservoirs within the same general location. The major criteria of geologic similarity are as follows: productive formation, regional geologic structure, depositional environment, hydrocarbon type, and trap type.

Figure 1.2
Supply Regions for the Gas Systems Analysis Model

DICTIONARY

POF:	Pacific Offshore
PON:	Pacific Onshore
SJ:	San Juan
RF:	Rockies Foreland
WI:	Williston
P:	Permian
MC:	Mid-Continent
AET:	Arkla-East Texas
TGC:	Texas Gulf Coast
GMW:	Gulf of Mexico-West
GMC:	Gulf of Mexico-Central
NP:	Norphlet
SL:	South Louisiana
WF:	West Florida
MF:	MAFLA Onshore
MW:	Mid-West
AP:	Appalachia
NA:	North Alaska
MD:	MacKenzie Delta
ALB:	Alberta
BC:	British Columbia
SI:	Sable Island
DI:	Distrigas
CP:	Cove Point
EI:	Elba Island
LC:	Lake Charles
MEX:	Mexico



The plays are defined to approximate specific exploration concepts and to indicate groups of reservoirs susceptible to similar development strategies. The U.S. play definitions in the NRG database have been compared with and correspond closely to the plays defined by the USGS in its national resource assessments. Additionally, Canadian play definitions in the NRG database have been matched with play definitions of the Geological Survey of Canada (GSC).

The undiscovered gas resource database in the system is more theoretical in nature than the known resource database. The undiscovered resource database was compiled using existing estimates of the undiscovered gas resource in North America, the most detailed of which is for the United States by the USGS. For Canada, undiscovered gas estimates are based upon the study by the Geological Survey of Canada. NRG Associates, a participant in the 1995 Department of Interior assessment of undiscovered resources, performed the crosswalk among the databases.

Other Input Data. Other information is needed to fully evaluate a gas reservoir's productivity and economics relative to other investments. Various capital and associated operating and maintenance (O&M) costs must be estimated as a function of specified technologies, location, and producing characteristics. Information on regional or national trends in drilling and completion costs, dry hole costs, and gas processing and waste disposal costs, along with other gas industry factors, is incorporated to reflect the dynamics of evolving capital and energy markets. Additional data on local, state, and federal tax structures, depletion, depreciation, and royalties, as they relate to gas producers and reservoir location and types, are used to provide reservoir evaluations on a project-specific discounted cashflow basis.

Current and advanced technology performance and cost assumptions and various policy conditions are defined as user-specified parameters. GSAM evaluates the evolution of decision parameters in response to constantly changing procedures and technologies as they are developed, tested, and implemented by operators in the field and as the character of the remaining resource base changes. It has been designed to directly evaluate and quantify the impacts of changing technology or policy conditions on the domestic gas system. It must therefore provide a flexible method for changing technology and policy conditions through user-specified input. This capability has been carefully designed into the various modules.

Finally, macroeconomic variables, infrastructure constraints and costs, and gas industry characterizations are required to fully represent the market within which investment decisions are made. These market parameters strongly affect the model's simulation of investments in new gas supplies, as well as end-use gas demand, by imposing realistic limits on regional supplies and inter-regional transportation volumes. These downstream factors are derived from historic industry practices and adjusted for fundamental changes in industry structure and recent regulations.

I.3 Basis of Analytical Assumptions in GSAM

In order to perform analyses on a reservoir-specific basis, a functional database containing various characteristics for each reservoir was required. The key characteristics of the reservoir database include gas, natural gas liquid, and oil production and reserves, horizontal permeability, total porosity, depth, initial pressure, water saturation, bottom-hole temperature, thickness, gas gravity, heating value, original gas-in-place, and area. A type curve methodology was selected as the modeling approach to be used by the system for representing a reservoir's production performance. Type curve models explicitly analyze reservoir performance based on actual reservoir properties and technology performance parameters.

In order to assure consistent analytical results from the model and to address R&D issues in a timely manner, several key aspects of data are handled by explicit specification in input files. These key data elements are derived from the literature, EIA publications, or the other available information. Availability of this data simplifies analytical procedures and drives modular design.

The types of input data contained in the model include information on rig utilization factors by region, gas demand factors by region, sector, and season, existing inter-regional pipeline capacities and operating costs, and drilling cost adjustment factors to account for varying market conditions (prices, rig rates, etc.). By making these specific input parameters, the system's modular design allows easy variation of these values, and these factors can also be updated, as required, to reflect future market starting conditions.

In addition to the direct starting inputs, the system requires data on future changes in drilling costs and/or efficiency, the market penetration of future technology advances and variations in competing fuel costs. These time- dependent variables are also designed as input to the modeling system to add to it's flexibility and versatility.

Model Analysis Logic

The North American gas resource is large, diverse, and widely dispersed, both geologically and geographically. Exploration and production (E&P) efforts encompass a broad range of activities and technologies. Many of these are explicitly designed for the specific characteristics of particular prospects or resources. Despite their disparate nature, conventional, offshore, tight, Devonian shale, coalbed methane, and sub-quality gas resources share common production obstacles. Resource data used in GSAM describe and allow modeling of each of these resources in a consistent and comprehensive manner. Further, the various exploration, development, production, and processing activities within and across various resources are also evaluated in a consistent analytical framework.

Exploration is evaluated on a fully-risked prospect basis. For exploration to be conducted, the expected value of the next discovery must exceed the full cost of finding hydrocarbons (including dry holes), and ultimately, developing and producing the potential discovered reservoir(s). Once a reservoir has been discovered, development and production from that reservoir must generate expected revenues to cover the investments, operating costs, and risks of development. These evaluations integrate detailed information on reservoir geology, technology applications, productivity, costs, and market prices -- the same information an operator uses in the field to select projects.

GSAM integrates these discrete decisions into a market framework. Investment decisions over time must be justified based on contemporary market conditions (e.g., capital and rig availability, wellhead prices) consistent with the supply and demand of gas and the availability of infrastructure in various regions. E&P activity that creates market imbalances (e.g., excess supply in a given region) must justify not only its direct extraction costs, but also the additional costs of transporting the gas to an end-user.

Based on aggregate activities in various supply and demand regions, the model equilibrates regional markets and prices over the forecast period. The downstream model also explicitly addresses seasonal demand fluctuations that influence gas infrastructure, storage, and utilization capacity and investment decisions. This comprehensive market framework ensures that the estimated impacts of R&D or tax policies on E&P activities appropriately and credibly reflect market realities.

The five main analytical modules, each of which can be run in a stand-alone mode, are as follows:

(a) Resource Module - transforms raw resource and reservoir data into fully characterized, reservoir-level data-bases. The module operates using several routines that evaluate available information and estimate missing data elements based on reasonable engineering and geologic default parameters.

(b) Reservoir Performance Module - estimates annual production volumes and costs associated with development of each known or potential producing natural gas reservoir characterized by the Resource Module.

(c) Exploration and Production Module - evaluates the exploration, develop-ment and production of the natural gas resource base over time as a function of contemporary market conditions and technology, economic, and policy assumptions. Gas prices can be exogenously input or calculated based on analysis using the Demand and Integrating Model.

(d) Demand and Integrating Module - evaluates demand for gas by region, sector, and season as a function of gas prices, population growth, economic activity, interfuel competition, and other regional and national factors. Creates input files for operating the linear program to balance supply and demand across a nationwide transportation network linking supply and demand regions.

(e) Production and Accounting Module - converts output from other modules to provide a full accounting of all exploration, drilling, completion, operations, and upstream activities. Output provides details on annual gas production, gross revenues, taxes, investments, operating costs, and operating profits.

I.4 Description of the Analytical Modules

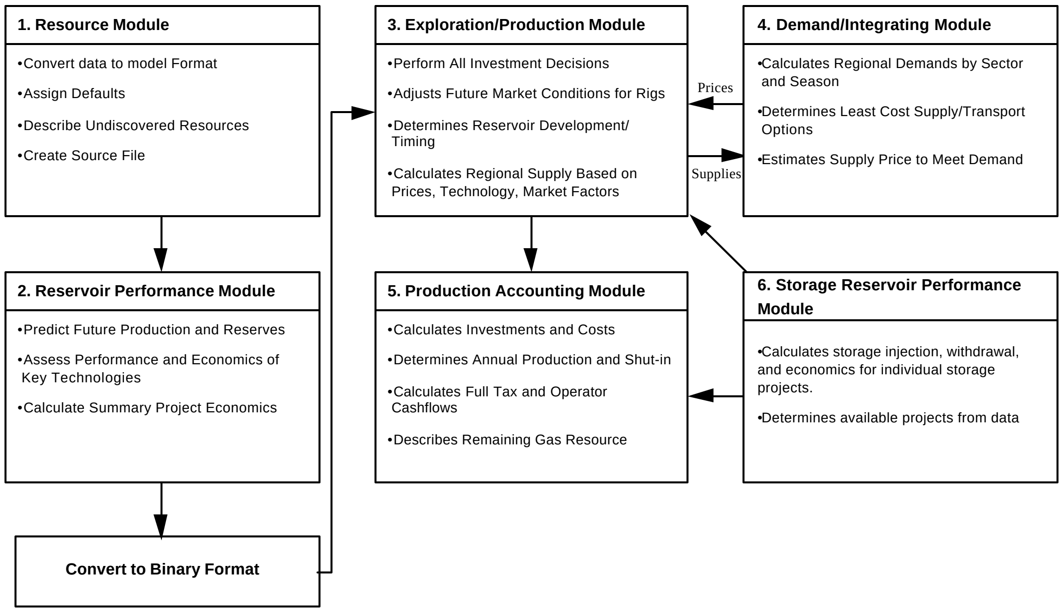
Figure 1.3 provides schematic overview of the major analytical components (modules). Each of these five components is summarized below.

Resource Module

The Resource Module converts resource data into reservoir-specific information in a format that the Reservoir Performance Module can use. All available raw resource data are incorporated into the Resource Module to be processed, analyzed, and validated into full reservoir descriptions. The module consists of more than simple screens, instead using the distribution of properties within plays to confirm information and estimate missing data. The full characterization of known producing, discovered but undeveloped, and undiscovered reservoirs in the United States and Canada draws on resource assessments available from various private and public sources.

Additional segments of these modules check for data consistency among reservoirs within plays by size and depth. This evaluation allows data default values to be determined and appropriately incorporated where reservoir data are inconsistent or missing. The completed reservoir descriptions are stored in one database for undiscovered/undeveloped reservoirs and another database for reservoirs for which initial development and significant production have occurred.

Figure 1.3
Major Components of the Gas Systems Analysis Model


1. Resource Module
•Convert data to model Format
•Assign Defaults
•Describe Undiscovered Resources
•Create Source File
2. Reservoir Performance Module
•Predict Future Production and Reserves
•Assess Performance and Economics of Key Technologies
•Calculate Summary Project Economics
Convert to Binary Format
3. Exploration/Production Module
•Perform All Investment Decisions
•Adjusts Future Market Conditions for Rigs
•Determines Reservoir Development/ Timing
•Calculates Regional Supply Based on Prices, Technology, Market Factors
Prices
Supplies
4. Demand/Integrating Module
•Calculates Regional Demands by Sector and Season
•Determines Least Cost Supply/Transport Options
•Estimates Supply Price to Meet Demand
5. Production Accounting Module
•Calculates Investments and Costs
•Determines Annual Production and Shut-in
•Calculates Full Tax and Operator Cashflows
•Describes Remaining Gas Resource
6. Storage Reservoir Performance Module
•Calculates storage injection, withdrawal, and economics for individual storage projects.
•Determines available projects from data

Undiscovered plays are split into two categories: hypothetical plays (plays with no current known reservoirs) and currently producing plays. For hypothetical plays, an analogous play in the model's database is identified and the characteristics of the analog play are used to describe the hypothetical play. Currently producing plays were matched to at least one play code in the database.

For each type of matching play, the same general methodology is used to estimate recoverable resource. In each play, a recoverable resource per reservoir is calculated for each reservoir size class in the play. A logarithmic function is then used to determine the relationship between the number of reservoirs in each size class within a play. The total recoverable resource in each play is determined by multiplying the number of reservoirs by the recoverable resource per reservoir in each size class and summing the recoverable resource across size classes for each play. The number of reservoirs is adjusted (while maintaining the mathematical relationship) until the recoverable resource in the database closely matches the resource reported by USGS or GSC. For hypothetical plays, since there are no known reservoirs, the recoverable resource is matched to the USGS or GSC estimate.

The play approach to exploration evaluation provides a comprehensive assessment of the resource at a level of detail appropriate to the system's analytical requirements. Because the model considers the full cost and potential of exploration activities nation-wide, a play-based approach (White 1981) is ideal for undiscovered resource assessments. Reservoir characteristics for undiscovered reservoirs are determined from average properties in known reservoirs described in the play. Using this relationship, the largest possible size class in that play is determined and the number of reservoirs is adjusted until the recoverable resource for the play matches the USGS or GSC estimate.

The database which results from this approach is closely tied to existing and accepted estimates of the undiscovered resource in the United States and Canada. Accurate play characteristics for currently producing plays and careful choosing of analog plays representative of each hypothetical play ensure that the data are consistent.

Where it was available, gas, natural gas liquid, and oil production and reserves for each reservoir are obtained directly from NRG's reservoir database. If there is no annual production for 1997 and no cumulative production for the reservoir in 1997, then NRG's field database is used to find production and reserves. Field production and reserves are allocated to each reservoir according to the number of major reservoirs in the field and the suite rank of each reservoir.

Also, if available, data for other variables characterizing each reservoir are taken directly from NRG's reservoir database. Some variables not contained in NRG's database are calculated by using other variables provided in the database (i.e., the total number of wells for each reservoir, the number of shut-in wells, the amount of productive acreage, and the amount of proved acreage). In some cases, if no values existed for variables, fixed defaults are assigned to these variables. Conditions and boundaries are set for variables to maintain consistency. These conditions are set by lithology, by state district codes, and by play code.

Reservoir Performance Module

The Reservoir Performance Module develops reservoir production response estimates and summary project economics based on the reservoir data output from the Resource Module and input on technology specifications, regional costs, state and federal tax requirements, and other assumptions. The production response estimates and project economics are subsequently used by other modules of GSAM.

The Reservoir Performance Module estimates initial production and conducts economic analyses that are used later in the Upstream Model to sequence natural gas resource development. This module incorporates reduced form reservoir models (type curves) that transform discrete reservoir properties and technology assumptions into a characterization of reservoir development and production profiles and ultimate gas recovery. These estimates are evaluated using appropriate costs and economic values to determine discounted net cashflows and project profitability.

The economically optimal technology scenarios are flagged and reservoirs that are never likely to be economic are excluded from further analysis. This preprocessor refines the list of reservoirs to the minimum set appropriate for the scenario being analyzed. This module design feature allows numerous technology scenarios to be evaluated prior to the demand for rapid scenario analysis, vastly reducing the analysis turnaround time.

The Reservoir Performance Module provides capabilities to analyze production response based on:

- Reservoir unit of analysis
- Pay grades with variations in key properties to capture heterogeneity
- Estimated ultimate recovery, reserves potential, annual production, and economics for alternative technology, economics, costs and investment requirements, and development options
- Independent analysis of variables related to technology and costs allowing sensitivity analyses
- Preprocessing of data to minimize analysis time and allow rapid sensitivity studies
- Capability to reanalyze economics and development of reservoirs in full discounted cashflow (DCF) model at end of run.

Central to the module is a series of type curves. These type curves estimate gas production from various reservoir settings based on average reservoir properties. The type curve estimation has the capability to analyze individual, uniquely described pay grades in each reservoir under a variety of development options (including conventional, infill drilling, and re-stimulation), providing production and pressure estimates for the entire life of each reservoir.

In selecting and designing a modeling approach, the large number of reservoirs to be analyzed in each run was an important factor. In addition, accuracy of results and the capability to analyze a variety of technologies was critical. The type curve approach was determined to be more appropriate than alternative modeling approaches, given data and analytical needs of GSAM. The goal of the design was to develop a consistent set of reservoir models that could predict future production under a variety of reservoir conditions, operating restrictions, and technology parameters. The individual models are completely consistent in their input requirements and output formats.

The gas production type curves estimate the impact of technology and development options on both rate and ultimate gas recovery. Like most type curves, they are single well models that estimate the production from a closed boundary system. The model type curves have been designed to estimate reservoir production under the following conditions:

- Up to three pay grades may be modeled, each with its own reservoir properties and technology application.
- Each well's production is partly determined as a function of proration rules that limit production to a fixed percentage of reservoir open flow. This limit is represented as a plateau rate initially (constant rate) until deliverability can no longer be maintained. After that break point, production declines (constant pressure) until the economic limit is reached or additional development is implemented.
- Infill/Recompletion episodes can be evaluated, either automatically or manually implemented.
- A variety of technologies can be considered including any that is represented in the gas flow equations, such as skin, flowing or abandonment pressures, drainage radius and shape, perforation efficiency, radial or linear flow (i.e., hydraulic fractures or horizontal wells) and wellbore radius.

The model tracks reservoir conditions such as pressure at the wellbore and at a potential infill location to allow evaluation of the relative economics of infill/recompletion at a later date. The output is formatted for use in developing individual pay grade evaluations, as well as reservoir level analysis of production and facilities requirements.

A total of six explicit type-curve models have been developed for the Gas Systems Analysis System:

- Conventional radial flow - single and dual porosity
- Linear flow (fracture stimulated or horizontal wells) - single and dual porosity
- Water drive

- Unconventional resources, with separate subroutines for wet and dry shales, and wet and dry coals.

Geopressured aquifers can be modeled by the radial, single phase model (with water as the liquid phase).

These type curves are explicitly designed to evaluate the performance and economic impacts of various technologies on future gas production. This is accomplished in the models by direct variation of parameters related to the exploration, drilling, completion, reservoir characterization, and production characteristics of individual reservoirs. These type curves were developed to characterize the explicit impact on well performance and costs of alternative technology scenarios, individually or in selected combinations.

Among the key performance variables that can be analyzed in the model are:

- Pay continuity factor (measure of pay contacted)
- Horizontal well lengths
- Fracture half-length
- Fracture conductivity
- Skin factor
- Wellbore radius
- System operating pressure.

These parameters simulate the direct impacts that technology advances will have on gas production from wells. They define the productivity potential of new R&D initiatives designed to increase and lower the cost of gas production. For this reason, the type curves were developed with the flexibility to evaluate each parameter independently or in combination with other improvements in reservoir performance, investment requirements, or risk reduction.

Individual investments and operating processes are uniquely assessed in the model to determine their cost to operators:

- Based on published sources at level of well, reservoir, prospect
- Adjusted/verified by vendor quotes for known costs
- Scaled up to reservoir/prospect

- Based on regional, rate/depth specific values, where appropriate
- Use commercial costs, not R&D level costs
- Are technology-specific
- Can be adjusted for market conditions.

The costs and investments used by the models and their sources include:

- G&G Costs (API survey, Bureau of Census)
- Lease Bonus data (published in trade press, Bureau of Census)
- Drilling and Completion (Joint Association Survey on Drilling Costs)
- Equipment (EIA)
- Stimulation (vendors)
- Operating costs (EIA).

The models characterize the capital and operating costs of finding and developing natural gas reserves as a function of location, technology used, and production performance. The models also contain routines for state income and production taxes, depreciation, depletion, and amortization schedules, and Federal income tax parameters.

The overriding principle of GSAM's decision-making is that all decisions are on the table - if gas prices drop to zero, then the model stops all drilling and curtails all production. There are no assumptions about market or sales inertia unless specifically put into the model.

Tax and policy assumptions characterize the impact on natural gas E&P activities and project economics of alternative public sector tax, leasing, regulatory, capital and other potential policies directly or indirectly affecting natural gas development.

Exploration and Production Module

Operator decision-making for upstream investments is performed for exploration, initial reservoir development, production, additional development, and additional sources (such as associated gas). The module evaluates pre-processed, project-specific production and financial summary data from the Reservoir Performance Module against user-specified decision criteria for contemporary market conditions.

The module ultimately determines the production from available reservoirs at projected gas price tracks. Again, the module considers options from the viewpoint of the operator, deciding whether to implement or defer various investment opportunities. For example, additional development is evaluated for investment options for infill drilling or recompletion of initial development wells to maintain reservoir deliverability.

The evaluations are all based on reservoir-specific calculations and consider the direct and secondary impacts of changing technology on future production, costs, and reservoir access.

Exploration. Economic evaluation of drilling new field wildcats in undiscovered reservoirs of various plays is performed if the expected value of full cost discovery is greater than the long-term wellhead prices provided by GSAM's Demand and Integrating Module (or at given price track such as AEO). Some adjustment may have to be made to the expected value of exploration to represent the long term need to replace reserves, even when prices are low.

The Exploration and Production Module estimates the costs and timing of successful exploratory wildcats as a function of the remaining resource base and contemporary exploration technology effectiveness and economics. It compares the fully-risked costs (inclusive of dry holes) of an exploratory well with the expected economic return from development and production of a discovery. The exploration methodology extends the Arps-Roberts exploration analysis concepts to incorporate more explicit technology effectiveness parameters. Because capital and drilling constraints directly affect exploration decisions, the Exploration and Production Module is linked directly to the Integrating Module. The wellhead gas price exploration decision criterion can be changed by the user to other criteria, such as reserve replacement, maximizing gas production, minimizing capital expend-itures, etc. These alternative decision criteria are quantified and incorporated into the expected profitability of a prospect.

GSAM uses a modified exploration play analysis approach based on White (1980). This approach incorporates the latest geologic data available from USGS and NRG. It also explicitly addresses the ability of technology to allow operators to preferentially test prospects based on uncertain reservoir property characterization.

The method incorporates a characterization of remaining prospects (as for White's geological model) based on empirical and subjective data on known reservoirs in each play. It replaces the Monte Carlo sampling scheme (as in White's exploration model) with an algebraic representation of testing prospects as a function of geophysical measurement accuracy and regional interpretations. The exploration model evaluates the full cost economics of drilling a series of prospects.

Based on the size, shape, and other detectable properties, as well as the probability of finding a reservoir based on its proportional area, the model estimates an expected net present value for a successful exploratory well. In traditional exploration models, one-time improvements in exploration technology usually result in permanent improvements in exploration efficiency. This approach reflects the reality that an improvement in technology that more accurately detects reservoirs of a certain size or trap type is only effective until the pool of those types of newly detectable reservoirs is depleted.

Initial Reservoir Development. The Exploration and Production Module develops reservoirs already discovered, but not yet developed, if the minimum required price on a sunk cost basis is lower than the expected wellhead price. The analysis in this segment is fully based on sunk exploration costs. Individual reservoirs that have been discovered as a result of previous exploration activities are analyzed to determine when operators would develop them. The evaluation considers the performance and economic evaluation completed in the Reservoir Performance Module for primary development to "normal" well spacing. Based on the timing of the development decision, technology conditions are also adjusted. Contemporary market conditions and tax structures are also considered in all development decisions. Development options depend on the type of resource and location, depth, and operating restrictions associated with each reservoir. Conventional reservoirs are evaluated with normal drilling and completion costs, varied based on regional and depth specific cost factors. Tight and shale reservoirs, as well as coalbed methane deposits, are assumed to require fracture stimulation for successful completions.

Once initial development has been initiated, the model schedules development based on the size of the reservoir. Initial development is phased in based on the number of wells required to completely develop each reservoir, with development drilling limited to one-fifth of the total wells required. If economic conditions change during the development period, drilling can be delayed or terminated, reflecting operator response to falling gas prices and increasing costs over time. The phasing of drilling is consistent with field operations where rig and infrastructure constraints limit operators' ability to develop various reservoirs.

Additional Development. The Exploration and Production Module incrementally develops reservoirs that have already been developed by conventional practices. The decision process, however, is at the pay grade level. Once developed to "normal" spacing, segments of the reservoir can be identified for exploitation using incremental development options. Two additional development decisions are possible: (1) infill drilling to some "close" spacing, and (2) re-stimulation of existing wells. These decisions are made based on an engineering evaluation of the status of wells at the time such incremental development is possible, relative to available technology and contemporary wellhead prices.

As with initial development decisions, all parameters are adjusted to the situation at the time of an operator's decision. Incremental economics are considered, ensuring that the additional development being analyzed offers an economic gain over continued operation of the reservoir under initial development status. Also, technology considerations are indep-endently evaluated in the module based on the market penetration at the time of the additional development investment.

The options for additional development of each reservoir are evaluated in the Reservoir Performance Module consistent with initial development criteria. This assures complete consistency of the analysis and correct incremental decision evaluation in the model.

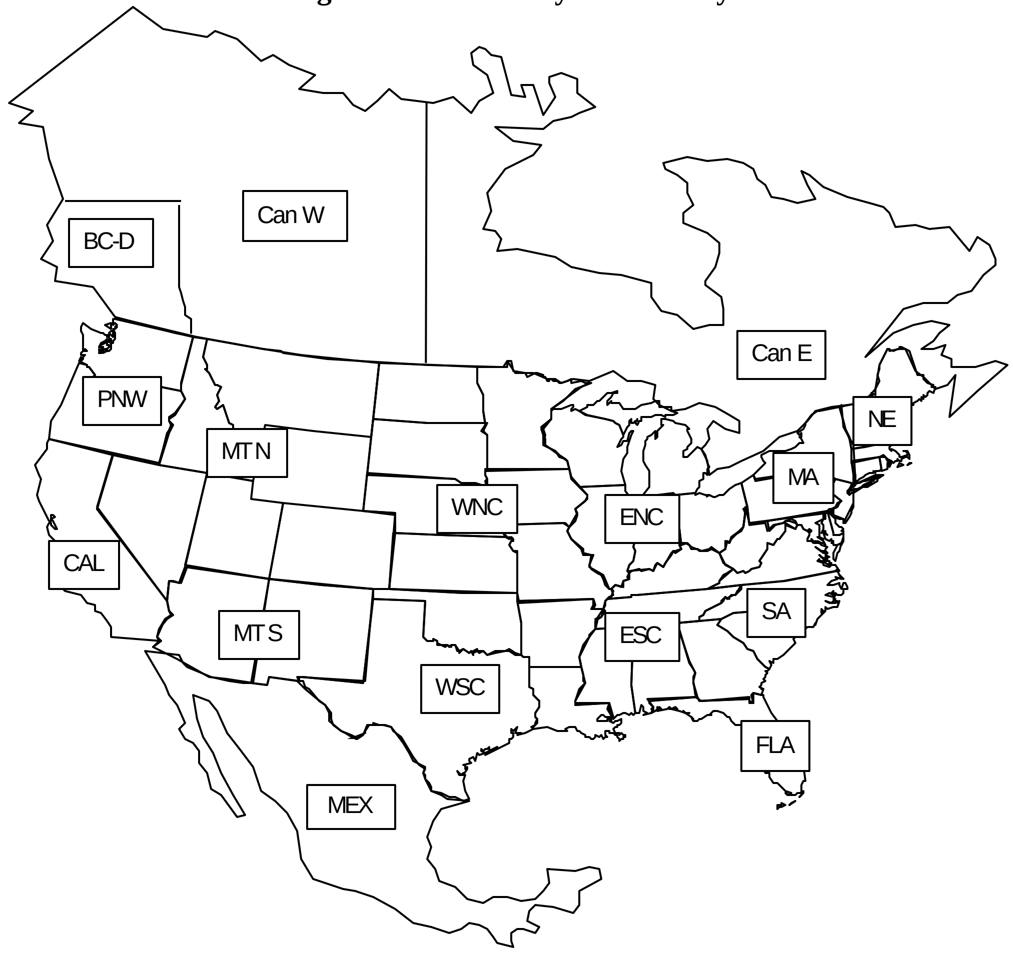
Developed Reservoir Production. The Exploration and Production Module produces or curtails wells producing from reservoirs that have completed initial development. This includes reservoirs already developed and on production as well as reservoirs developed over time. The module evaluates annual reservoir and pay grade production and evaluates annual shut-in decisions based on operating costs, royalties, and taxes on that production relative to available wellhead prices provided by the Demand and Integrating Modules or the user. Production is shut-in when total costs exceed total revenue from a well or pay grade.

Production from these wells may be curtailed for several years if wellhead prices are falling but later rise. Once prices increase to a point where revenues exceed total cost of operations, production can be reestablished. However, pay grades and wells are assumed abandoned after a shut-in period of three years.

This segment of the model provides estimates of the productive life span of individual wells. The evaluation is simple and direct and consistent with other decision evaluation methods in the modeling system. The shut-in decision is an independent assessment done on a well-by-well basis under conditions at a particular point in time.

Demand and Integrating Module

Figure 1.4
Demand Regions in the Gas Systems Analysis Model



The downstream components characterize the transportation and end- use market segments of the natural gas industry in sufficient detail to represent the dynamics of market supply and demand interaction and price formation. The structure is designed to allow users to assess benefits of gas supply research and development initiatives in light of operational, demand, and ultimate supply price impacts. In addition, the downstream model allows the user to assess the implications of changes in downstream policies on supply-related activities. The downstream model has the following major characteristics:

Regional Demand. Gas demand is presented in 16 regions or nodes. In the United States these are based on the nine major census bureau regions, subdivided in several cases to provide additional detail. In Canada, three regions were created to simulate the Canadian natural gas market. The demand regions used in the model are illustrated in Figure 1.4.

Sectoral demand. Within each region, gas demand is represented in four end use sectors: residential, commercial, industrial, and electric generation. Both industrial and electric generation sectors are further subdivided into market segments based on alternative fuel use in order to simulate gas and oil competition. The model simulates competition between gas and distillate, low-sulfur residual fuel oil and high-sulfur residual fuel oil. In addition, the electric generation component includes a sub-model for estimating the level of gas-fired generation capacity and generation that would occur at different price levels for natural gas.

Seasonality of demand. Gas demand is represented in four seasons: a 5 day peak period, 26 day shoulder period, 90 day winter season and 244 day summer season. During summer, storage becomes a demand sector; in other three periods, storage is treated as "local" supply source, much like an alternative fuel, for meeting demand.

Transmission. The transmission network of GSAM consists of around 80 bi-directional links (Figure 1.5). The links connect gas supply nodes with other supply nodes and ultimately with gas demand nodes. Each link is characterized by maximum capacity, fixed costs, variable costs, and fuel. The model endogenously expands capacity on links when economically justified. Distribution costs are treated as a margin added onto the delivered city-gate gas cost.

The downstream model operates by generating gas demand curves for each region. The transportation network integrates the gas demand curves with the gas supply curves by means of a linear program that minimizes the total cost of meeting demand, inclusive of transmission and gas costs.

The underlying assumptions concerning the existing structure, future expansion, and general operation of the downstream gas market which the model is designed to represent fall into three categories:

- Supply/demand equilibrium
- Regional pricing
- End use pricing.

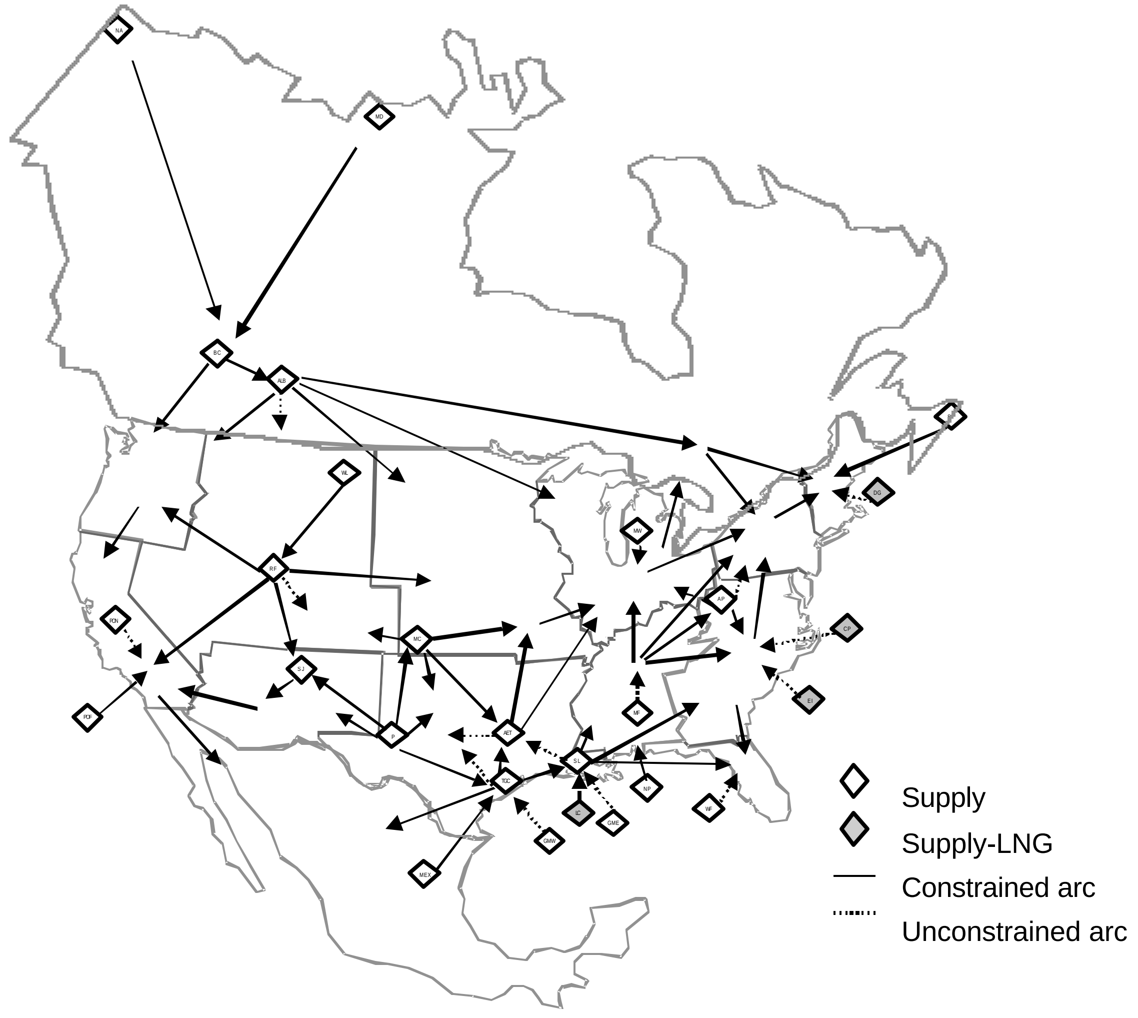
The model assumes that the gas market is workably competitive and that gas prices will adjust upward or downward to balance supply and demand, consistent with economic theory. In today's highly integrated marketplace, producers now have multiple opportunities to reach markets, and buyers now have multiple routes by which to purchase and transport gas.

The broad result is at the gas market where prices are determined by the interaction of supply and demand and competition. Gas competes in this market with other fuels and conservation. In addition, gas from different supply regions compete markets in demand regions. Gas-on-gas competition has a major bearing on the price of gas in the national marketplace.

The model is structured to reflect the competitive nature of the market and to find the price that will bring gas supply and demand into equilibrium.

The downstream model also reflects a dominant feature of today's gas marketplace -- the regional pricing of gas in reference to an overall national market price. The New York Mercantile Exchange (NYMEX) futures contract is traded at Henry Hub in south Louisiana. Regional prices in the United States are set in reference to the Henry Hub price through a complex interaction of transportation margins and pipeline capacity.

Figure 1.5
Transportation and Pipeline System of GSAM



In consuming markets, the clearing price of gas is determined in two ways: it is capped by the price of the competing alternative fuel and by the cost of gas (inclusive of transportation) from competing gas supply markets. In supply markets, the clearing price is based on a transportation netback from the consuming markets; supply will flow along the path that provides the highest netback to the producer.

The model also reflects competition between pipelines in keeping with the structure of today's market. Pricing differentials tend to reflect the cost of transportation; during peak times, this differential is based on the full costs (so-called 100 percent load factor) of transportation, but in the off-peak periods, the differential falls to the variable costs of transportation (essentially the cost of fuel and variable O&M). Gas will fill up the low-variable-cost pipelines first and then the higher-variable-cost pipelines. Pipeline capacity will be added when the full cost of transportation plus gas prices can still clear the market at higher levels of demand.

The model is designed to represent a key feature of today's market -- that is, the fact that the marginal users of gas have seen the average margin charged to them decline relative to other customers since they have several alternative fuel sources. Where these users can be kept on the gas system and still make a contribution to fixed costs, regulators have allowed the margins to shrink. Thus, the cost to the marginal user will tend towards the wellhead price plus the variable costs of transportation. In the winter, the marginal user will tend to see a full cost of transportation. The model represents this by allowing marginal customer groups -- industrial boilers and electric generators -- to buy gas based on variable costs. Other customer groups see a price with full fixed and variable costs, including distribution margins.

The Integrating Module uses results from the Demand Module and the Exploration and Production Module to equilibrate annual gas prices and sales volumes over the entire period of analysis. The regional assessment of supply and demand is reconciled to determine inter-regional gas flows and resulting regional gas prices, from wellhead to end-use. Linear programming techniques solve for gas price and sales volumes against physical capacities and economic constraints among and within the 26 supply regions, 16 demand regions, and over 80 transportation links.

Production and Accounting Module

The Production and Accounting Module provides summary output on details not reported elsewhere in the modeling system. It has been designed to answer key policy and planning questions and provide comprehensive output on the results of model analyses. This module is designed to construct final analytical results based solely on data and assumptions developed used in the other modules.

The Production Accounting Module consists of a series of routines that read and sort output files from the Exploration and Production Module and the Reservoir Performance Module. The purpose is to conduct a final accounting of production, revenues, taxes and royalties, operating costs, and investments once drilling decisions have been evaluated and timed. The analytical structure is consistent with the economic evaluation methods developed for the Reservoir Performance Module.

I. 5 Overview Summary

The modular design of GSAM provides for maximum flexibility in analyses in estimating future supply, demand, and market conditions for natural gas.

The consistent evaluation of individual reservoirs and technology performance based on unique rock and fluid properties, as well as locational factors, significantly enhance the model’s capabilities in assessing reservoir performance. Decision-making in exploration and production is consistent from an operator’s perspective, leading the model to a robust market-based simulation of producer choice. Demand, by region, season, and sector, is calculated in sufficient detail to allow for major or minor occurrences in the downstream market. The upstream and downstream integration creates realistic regional prices, balancing supply and demand utilizing a transportation and storage network. The model, by producing a market-based, fully integrated assessment of natural gas potential in North America, is a unique tool for analyzing both business and public policy issues.

II. General GSAM Assumptions

GSAM is a reservoir level model (accumulations for undiscovered resources) and application of new technology is modeled at individual reservoir/accumulation level exactly in the same manner, as an operator would use the technology. Such level of sophistication is inherent in GSAM modeling framework and modules since it operates at the reservoir/accumulation level and contains performance routines consistent with the resource type level analyzed.

II. 1 Natural Gas Resources and Reserves

GSAM uses USGS estimates of undiscovered natural gas resource for U.S. basins. Canadian Gas Survey of Canada (GSC) is used for undiscovered resource estimates of Canadian basins. USGS provides F5, F95 and mean estimates of undiscovered conventional as well as unconventional natural gas resource by geologic play. Mean play level resource estimates are utilized in GSAM to determine total productivity from the geologic basins. The 1999 release of NRG Associates database is used for proven reserves in Canada and U.S. In addition, recent publications from Canadian Association of Petroleum Producers (CAPP), Canadian Gas Potential Committee (CGPC), Energy Information Administration (EIA), Minerals Management Service (MMS, for offshore Gulf of Mexico), Oil and Gas Journal, and USGS open file reports are researched to improve and complete (wherever needed) the resource database. The key references are:

For U.S. basins:

1. 1995 National Assessment of United States Oil and Gas Resources - Results, Methodology, and Supporting Data - United States Geological Survey (USGS), Digital Data Series DDS-30, DDS-35 and DDS-36, Published in 1996.
2. 1995 National Oil and Gas Assessment and Onshore Federal Lands - United States Geological Survey (USGS), Open File Report 95-75-N, January 1998.
3. U.S. Crude Oil, Natural Gas, and Natural Gas Liquids Reserves Report, Energy Information Administration, www.eia.doe.gov, various reports.
4. The Significant Oil and Gas Fields of the U.S., 14th Update, NRG Associates, August 1999 release.

For Canadian basins:

1. Natural Gas Potential in Canada - A Report by the Canadian Gas Potential Committee, 1997.
2. Coalbed Methane: A Comparison Between Canada and the United States, Geological Survey of Canada (GSC), Published in 1995.
3. The Significant Oil and Gas Pools in Canada, NRG Associates.

II.2 Definitions of Various Categories of Resources in GSAM

Experience shows that initial estimates of the size of newly discovered natural gas reservoirs and fields are usually too low. As years pass, successive estimates of the ultimate recovery of fields tend to increase. The term "reserves growth" refers to the typical increase in estimated ultimate recovery that occur as natural gas fields are developed and produced. The following example for a particular field will help explain the nature of reserve growth. A large natural-gas field located in Texas was discovered in mid 1940's. In year 1977, its ultimate recovery was estimated to be 2.1 Tcf. One might think that after some 20 years of development and production, the resource potential of a field would be well understood. However, by 1991 the estimated ultimate recovery of this field had increased to 3.1 Tcf. Reserves growth over the 15-year period totaled 1.0 Tcf and showed no sign of stopping.

Historical data suggests that most of the reserves growth comes from existing fields. From 1989 to 1993, reserve growth from existing fields contributed to about 1 Tcf/year to U.S. proved reserves, whereas new-field discoveries added only 1 Tcf/year. In recent years, USGS estimates that reserves growth from existing fields has contributed far more to U.S. proved reserves than new field discoveries.

NRG Associates database is consistent with proven reserves estimates reported by EIA which stands around 140 Tcf for Non-Associated gas (Associated gas proven reserves are around 25 Tcf, bringing the total proven reserves to around 165 Tcf) as of year-end 1997 (the start year of GSAM). However, as historical data suggests, the proven reserves number does increase substantially primarily due to reserves growth in existing fields. GSAM does capture this phenomena primarily because it operates from Original-Gas-In-Place (OGIP) standpoint and not from proven reserves standpoint in terms of forecasting production estimates.

Practical definition of reserve growth includes the following four components

1) New reservoir discoveries in old fields (to connect by-passed zones, i.e. field/reservoir merger)

2) Revisions in old fields (i.e. upward revisions of proven reserves calculations based on production experience and changing relations between price and cost)
3) Extensions in old fields (i.e. physical expansion of fields by areal extensions and development of new producing intervals or reservoirs)
4) Enhanced recovery techniques applied in old fields (i.e. improved recovery resulting from application of new technology and engineering methods)

GSAM resource base does include undiscovered resource, reserve growth resource and proven reserves. The reserves growth resource estimate includes growth due to new drilling in old fields to connect by-passed zones, revisions and extensions. All resource and reserves estimates in GSAM are based on non-associated gas basis. Associated gas production in GSAM is obtained through external estimates and is added to non-associated gas production on an annual basis to get total gas production. Definition of proven reserves in GSAM includes recovery in existing formations resulting from application of advanced technologies and engineering methods. Reserve growth (or appreciation) potential in Canada is defined on a pool basis and hence only represents extension and infill potential in existing pools. New pool reserves growth potential is classified as undiscovered. Due to this reason, reserves growth estimates for Canada are much lower, because this component is actually grouped in the undiscovered resource.

II. 3 Federal Lands Issues

The Federal Government is currently the largest owner of oil and gas resource in the United States. Of these resources, a large percentage is restricted from use and production based upon governmental policy, specifically moratoriums imposed on drilling/production in the OCS areas, and leasing and development permitting delays on onshore areas. If these restrictions were eased or removed, a large portion of the resource on Federally owned lands could be produced. As a result, the actions that the government could potentially take with respect to these resources can have a vast impact in all aspects of the oil and gas industry. With these factors in mind, DOE HQ has completed studies to understand historical as well as plausible future Federal land leasing activities to ensure accurate prediction of impacts on production, reserves, cashflow and related employment levels due to federal lands leasing trends.

Significant portions of the domestic natural gas and petroleum reserves, particularly estimated undiscovered reserves, are located on Federal lands. GSAM assumes that, on average oil and gas development, activities on Federal lands are constrained (about half of private lands) due to a decline in access to these resources. These include policy and environmental issues such as OCS moratoria, wilderness land designation, NEPA rules, National Parks, etc. In addition to access, delays in the approval of development plans can severely reduce the producability of natural gas from these lands.

GSAM's federal lands availability assumptions are based on US Department of Energy recent studies. Through our interactions with various Government agencies including Bureau of Land Management (BLM), Forest Service, Minerals Management Service, industry associations, and experts in this field, we have developed a reasonable understanding of the resource underlying Federal lands, and of the issues affecting the access and development of these resources.

GSAM has an elaborate modeling framework to characterize the natural gas and petroleum resources on public lands, characterize the access and development scenarios, and quantitatively assess the impact of changes in public policy towards development of these resources. For undiscovered resource, raw data from USGS on federal percent by play is used to distribute recoverable resource on private and federal lands by geologic play. This helps in providing forecasts of production from federal and private lands separately under different resource availability and technology penetration scenarios. For producing reservoirs, GSAM database can distinguish production from federal and private lands at individual reservoir level.

II. 4 Future Environmental Compliance Requirements

An important consideration in assessing the potential of North American natural gas supplies to satisfy future market requirements is the impact that environmental considerations will have on the costs of future supplies. While more stringent future environmental requirements could increase the costs of E&P operations, thereby reducing the supply of natural gas available at a given price, future technological advances could reduce the costs of compliance.

For the purpose of estimating the impacts of potential future regulatory requirements (to be input in GSAM), our base case encompasses a plausible range of future environmental regulation based on our interactions with U.S. DOE, and EPA. The base case assumes a balanced, risk-based approach to future environmental regulation and other environmental initiatives such as technological advances. Environmental compliance requirements continue to increase in the future. However, future requirements consider the environmental risks of the regulated activities, the effectiveness and environmental benefits of the compliance requirements, and the total cost of the compliance requirements. The environmental compliance costs developed for the base case are above current costs.

GSAM's approach for estimating future environmental compliance costs starts with a comprehensive review of the projected environmental initiatives and emerging environmental compliance technologies in each of the environmental issue areas listed below:

1) Drilling and Drilling Waste Management
2) Production Waste Management
3) Air Emissions
4) Discharges and Chemical Releases
5) Produced Water Management
6) Remediation
7) Underground Injection Control

Within each environmental issue area, exploration and production activities likely to be affected by future regulatory initiatives or technology development are identified. Altogether approximately forty environmental issues or industry E&P activities are considered. For example, in the Drilling Waste Management issue area, the industry E&P activities affected by regulatory compliance issues or environmental compliance technology include: onshore drilling waste management; use of synthetic drilling fluids; and drilling in wetland areas.

Because forecasting future environmental compliance requirements is a highly uncertain exercise, a probabilistic or "expected value" approach is used to calculate an incremental cost of compliance or incremental technology-based cost savings for each of the E&P environmental issues considered. Probabilistic estimates account for future uncertainty because future costs are represented by the sum of the probability-weighted cost of alternative regulatory or technology scenarios.

- A unit cost of compliance or unit cost savings is calculated for the case using the best available sources of costing data.
- A probability of occurrence and year of implementation is estimated for the case appropriate to the philosophy underlying each case. The sum of probabilities for all the activity equals 1.0.

- The unit cost of compliance or unit cost savings is multiplied by its probability. The probability-weighted costs for the case are summed to obtain a final “expected value” compliance cost for each industry E&P activity.
- Future incremental compliance costs for all industry activities are summed by applicable year of implementation to provide the total “per well” incremental compliance cost for a given year. The assigned year of implementation determines the year in which the incremental cost is applied.
- The regulatory compliance cases require incremental compliance costs (additions to present costs) while the technology R&D cases produce cost savings to industry (decrements to present costs). Costs associated with technology development scenarios are shown as negative.
- In the final step of the cost analysis, the total incremental compliance costs for each case are input to GSAM as capital costs or operating costs applied to gas wells and specified by region, depth interval, resource type, or specified reservoir parameters.

II. 5 Exploration and Production Technology

Future production of natural gas from various geological basins in the U.S. and Canada is highly dependent on technology improvements. Historically, technology has played a major role in North American gas supplies. A majority of technology levers are explicitly captured in GSAM. Some of the key parameters are:

1) Improved efficiency of drilling, equipment, and operating costs - GSAM has current costs and separate decline factors for drilling, facilities and operating and maintenance costs.
2) Improving success rates (i.e. reducing the number of dry holes) - GSAM has dry hole rates for development and exploration wells that can be varied by resource types.
3) Increasing recovery from existing reservoirs through hydraulic fracturing - Fracture half-lengths and hydraulic conductivity values can be specified for reservoirs located in different regions. GSAM has separate type curve modules for assessing production potential from naturally fractured reservoirs and reservoirs with induced fractures. Skin factors by resource types can be specified.
4) Horizontal wells can be specified in GSAM for calculating future production from different basins. For undiscovered and undeveloped resource, selection between using horizontal wells or fracturing in vertical wells, is based upon field size class cut-off and investment efficiency calculations.
5) Revealing new areas and types of resources for exploitation through innovative geologic and engineering concepts - GSAM has parameters in its input files which reflect use of improved seismic

and reservoir monitoring techniques. Parameters are available by resource type, which indicate the percentage by which exploring hydrocarbon is easier in bigger field size classes because of efficient seismic techniques and the fact that they encompass bigger areal extent.

II. 6 Frontier Resources

GSAM contains view on frontier resources located in Canada and Alaska North Slope. In Canada, GSAM has supply curves for Mackenzie Delta region which start to produce in year 2010. Alaska North Slope is assumed to come on-line in year 2007. Mackenzie Delta is assumed to be delayed primarily because of local regulatory approval process which is time consuming in Canada. The supply curves are created based on published information in various trade publications (such as Oil and Gas Journal, Natural Gas Week, Gas Daily, Transportation and Storage Hub etc.). Currently, GSAM does not have reservoir level databases for the frontier supply resources because of lack of data. As data become available new reservoir/accumulation level input databases can be created. In addition to the supply sources, GSAM does have the capability of transporting gas from these frontier locations to potential markets through pipeline links.

In addition, to these two frontier resources, GSAM has the capability of adding new frontiers in its modeling framework. Such frontiers that can be added at later stages are Deep Gas, Ultra Deep-water Gas, Landfill gas, Grand Banks etc.

The share of Canadian natural gas, in the domestic energy mix of the U.S. is estimated to continue to be significant in the future as well. Although the Western Sedimentary Basin of Canada (comprising of basins in Alberta, British Columbia and Saskatchewan) holds the bulk of the natural gas reserves in Canada, a number of new and frontier regions are emerging as well. These include the Eastern Canada frontier basins including Sable Sub-Basin in Nova Scotia, and Grand Banks and Labrador Shelf in Newfoundland, and the Northwest frontier basins in the Mackenzie Delta and Beaufort Sea areas along the Northwest Territories and Yukon. New data-sets (supply curves) can be created and their impact on overall North American supply, demand balance can be studied.

III. Clarification on GSAM User's Guide

In November 2000, we responded to the questions raised by NETL on GSAM documentation. The responses are divided by GSAM module type namely Resource Module, Reservoir Performance Module, Exploration and Production Module, Demand and Integrating Module and Storage Reservoir Performance Module. No clarification was needed for the Production Accounting Module of GSAM.

III.1 Resource Module

Discussion of the creation of offshore reservoir files: The discovered database of GSAM has been updated based on the 14th update version (production data through 1997) of the Significant Oil and Gas Fields of the United States from NRG Associates. The following procedure describes the method that is currently implemented in the GSAM database for stacking reservoirs in the offshore fields of the Gulf of Mexico (GOM) regions:

1. Aggregate GOM offshore reservoirs into 12 geographic plays (and not geological play) based on water depth:

Table 3.1: Water Depth Aggregation

Water Depth			GSAM PLAY CODE		
(meter)	(feet)	Average (feet)	GOM-C	GOM-W	GOM-E
0-60	0-196.8	98.4	9901	9905	9921
60-200	196.8-656.2	426.5	9902	9906	9909
200-900	656.2-2952.7	1804.5	9903	9907	9923
>900	>2952.7	5000	9904	9908	9910

2. Aggregate reservoirs located within the same NRG cluster/field into a single reservoir according to the following guidelines:

   - Reservoir acreage is set equal to the largest reservoir acreage in the field

   - Well depth is set equal to the deepest reservoir well depth in the field (this is because, the same well is intersecting different zones and hence, finally in terms of designing the surface facilities the actual depth, which will also be the deepest, needs to be used. Drilling costs should also be calculated in the similar manner. It is the actual drilling cost and hence does not hurt overall costs)

- Water depth is set equal to the deepest reservoir water depth in the field (this is because, the same well is intersecting different zones and hence, finally in terms of drilling cost calculation, the actual water depth, which will also be the deepest, needs to be used).

- Field level data for each individual reservoir is added to give the Original Gas In Place (OGIP), gas production, oil production, NGL production, total number of wells, number of active wells, and number of inactive wells for the aggregate reservoir

- Data from each individual reservoir is averaged on a volumetric basis to give permeability, porosity, initial gas and water saturation, initial pressure, gas specific gravity, bottom hole temperature, CO_2, N_2, and H_2S concentration, and gas Z-factor data for the aggregate reservoir. This ensures that larger reservoirs are represented proportionately

- Other reservoir properties such as area and depth are set equal to the properties of the largest acreage reservoir in the field.

This updated database is the first version of GSAM database (GSAM Version 2000) that implements reservoir stacking for offshore gas fields in the Gulf of Mexico (GOM) regions. The GOM offshore gas reservoirs located in the same field are stacked together and stored as one field in the GSAM database. The updated GSAM database of GOM regions consists of 783 fields from 1356 individual gas reservoirs reported by the NRG Associates.

In Gulf of Mexico, geographic plays (with average water depth for the play as shown in Table 1) are chosen because of extensive stacking of reservoirs. One well may intersect different reservoirs in different geologic plays and therefore, geologic plays are not used anywhere in the Gulf. Reservoir Stacking in different onshore locations have been proposed to NETL. We feel it is important to implement stacking algorithm in Texas Gulf Coast, Rockies Foreland, Permian and ARKLA – East Texas regions of GSAM as suggested in our March 2000 report to NETL.

Play level resource estimate discussion: NRR (i.e. number of undiscovered accumulation) is calculated in the resource module (specifically in "undisc.exe" for U.S. conventional reservoirs and so on) of GSAM.

It is calculated based on Arps and Roberts methodology and considers number of discovered reservoirs from NRG Associates by play in assessing the NRR's by field size class and play for undiscovered resource. For each play, undiscovered resource estimates are reported in various input files for the resource module (such as: undisc.dat for undiscovered conventional resource, unconv.dat for unconventional resource and undoff.dat for offshore resource).

How is the field size class table used in the module? In GSAM accumulations are aggregated into the field size class definition (in terms of average recoverable resource) as specified in Table A-15 of the User's Guide.

III.2 Reservoir Performance Module

Procedure for creating the GSAM99 file: GSAM99.GSM contains pseudo discovered reservoirs developed to account for shortfall in production and reserves that occurs after the Reservoir Performance Module is run on NRG reservoirs. Due to lack of current reservoir pressure data in NRG database, the production forecast from the database may not be accurate for few reservoirs. Hence, pseudo reservoirs (generally one reservoir per GSAM region where shortfall exits) are created to account for the shortfall. These reservoirs are created from similar reservoirs in the region with similar reservoir rock and fluid properties and production histories. New pseudo plays (altogether 8) are created to accommodate these reservoirs. These plays are named IC01, IC02, …IC08.

What do we mean by reservoir decisions? Reservoir decisions as shown in Table B-13 of the User's Guide simply means all the possible options of a reservoir that are analyzed in the Reservoir Performance Module of GSAM. For a GSAMID, there are altogether 18 options (3 pay grades, 3 technology types and 2 technologies) that are assessed in the Reservoir Performance Module. These options (or decisions) are processed in the Exploration and Production Module simultaneously to select the most economic option. These options/decisions are rank ordered based on MASP.

III.3 Exploration and Production Module

How gas prices are created in GSAM using EIA data? EIA publishes Annual Energy Outlook (AEO) every year which contains average gas price at the L-48 wellhead. EIA also publishes regional wellhead prices in AEO supplements every year and tabulates the data by NEMS regions. The 1999 AEO supplement can be found at EIA's website and is used to create GASPRC.* file (website for ' 99 supplement:http://www/eia/doe/gov/oiaf/supplement/sup99g.pdf). NEMS region are mapped with respect to GSAM region and particular year $ values are converted in 1995$ before entering the numbers in GASPRC file. Remember, all values in GSAM currently are in 1995 dollars.

What is CAPP? Where do basin differential come from? CAPP is the Canadian Association of Petroleum Producers (CAPP) which represent around 170 companies whose activities focus on exploration, development and production of natural gas, natural gas liquids, crude oil, synthetic crude oil, bitumen and elemental sulphur throughout Canada. CAPP member companies produce approximately 95% of Canada's natural gas and crude oil. CAPP's 110 associate member companies provide the broad range of services that complete the infrastructure of this country's upstream crude oil and natural gas industry. CAPP publishes a handbook called "Statistical Handbook" every year which contain a variety of upstream natural gas data (such as drilling costs, wells drilled, gas prices, production, operating wells, reservers etc.). GSAM's Canadian data is primarily based on CAPP's Statistical Handbook. This handbook has been published since 1955 in Canada and is a key source of upstream petroleum statistics. Please contact Stephen Rodrigues, Technical Analyst, CAPP at (403) 267-1107 (website is www.capp.ca) with questions or comments on the publication contents.

Basin differential in GSAM comes from published wellhead prices in different regions in the US (which are obtained from EIA's AEO) and Canada (both Alberta and British Columbia which are obtained from CAPP). Basin differential is typically defined as the difference of wellhead gas prices between different producing regions.

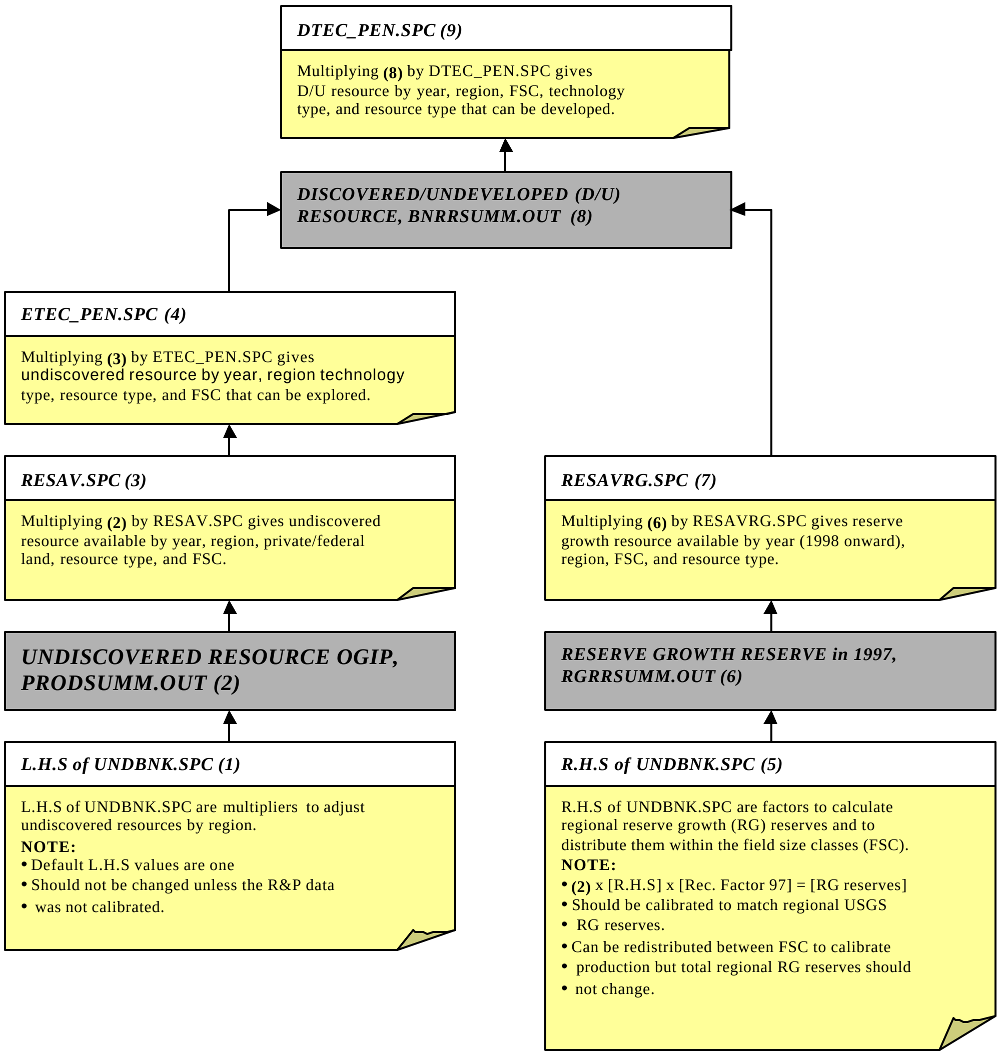
How different lengths of runs can be performed in GSAM? Different lengths of E&P runs in GSAM can be performed by changing years in GEN_TML.SPC file. Please remember to create entries for the corresponding years in other related files. For example, if the user plans to initiate a run upto 2020, entries in etec_pen.spc, dtec_pen.spc, resav.spc, resavrg.spc, etec_fed.spc, dtec_fed.spc, gasprc.new, and env_dat.spc should cover upto year 2020. If values are specified in these files are upto 2020 (i.e. less than the specified year of 2020) and the user intends to run the E&P model only upto 2000, only a simple change in the GEN_TML.SPC file is all that is needed. The user will specify in the year section of GEN_TML.SPC file, "1997, 1998, 1999, 2000". This way, the E&P model will only run upto year 2000.

What do you mean by volume when RG multiplies are discussed? Whenever we mention Volume it means gas resource. In GSAM, the RG multipliers and RESAV.SPC file works in the following manner.

Figure 3.1 shows in detail the inner workings of different files associated with reserves growth and reserves availability.

Figure 3.1

Input Files Involved in Resource Movement in GSAM



DTEC_PEN.SPC (9)
Multiplying (8) by DTEC_PEN.SPC gives D/U resource by year, region, FSC, technology type, and resource type that can be developed.
DISCOVERED/UNDEVELOPED (D/U) RESOURCE, BNRRSUMM.OUT (8)
ETEC_PEN.SPC (4)
Multiplying (3) by ETEC_PEN.SPC gives undiscovered resource by year, region technology type, resource type, and FSC that can be explored.
RESAV.SPC (3)
Multiplying (2) by RESAV.SPC gives undiscovered resource available by year, region, private/federal land, resource type, and FSC.
RESAVRG.SPC (7)
Multiplying (6) by RESAVRG.SPC gives reserve growth resource available by year (1998 onward), region, FSC, and resource type.
UNDISCOVERED RESOURCE OGIP, PRODSUMM.OUT (2)
RESERVE GROWTH RESERVE in 1997, RGRRSUMM.OUT (6)
L.H.S of UNDBNK.SPC (1)
L.H.S of UNDBNK.SPC are multipliers to adjust undiscovered resources by region.
NOTE:
• Default L.H.S values are one
• Should not be changed unless the R&P data
• was not calibrated.
R.H.S of UNDBNK.SPC (5)
R.H.S of UNDBNK.SPC are factors to calculate regional reserve growth (RG) reserves and to distribute them within the field size classes (FSC).
NOTE:
• (2) x [R.H.S] x [Rec. Factor 97] = [RG reserves]
• Should be calibrated to match regional USGS
• RG reserves.
• Can be redistributed between FSC to calibrate
• production but total regional RG reserves should
• not change.

III.4 Demand and Integrating Module

How convergence checks can be performed in GSAM? Convergence check in GSAM is performed after all the 10 integrating passes are completed. After all the 10 passes are completed, user need to calculate the difference in wellhead gas prices for every region and time period. If difference in gas prices are decreasing over time and are within 10 cents per Mcf for all regions and time period, it is considered that convergence is achieved.

What is dual price? Dual price (or shadow price) is defined as the price that the market is willing to bear for an additional unit of product (in this particular case natural gas) for consumption. In GSAM, natural gas market prices are calculated based on the concept of the dual price. Some general comments about dual price calculation in GSAM are as follows:

1. For a particular, region, year, and season, "How much better would the natural gas grid be with one additional unit of gas"

2. These calculations taken into account all regions, years, and seasons simultaneously

3. These calculations reflect the value of each potential activity that could be performed relative to adding one unit of gas to arrive at a "marginal activity". Example of these activities include:

    - Adding pipeline capacity

    - Increasing the level of gas extracted from storage

    - Decreasing the demand for gas in a particular sector, etc.

III.5 Storage Reservoir Performance Module

How are the two input files created for SRPM? The two input files for SRPM (STODIS.STO and STOUND.STO) are created from various publicly available literature. For active storage sites (reservoirs) we have used the following sources:

1. American Gas Association's (AGA) 1999 release of "Underground Storage of Natural Gas in the U.S. and Canada"

2. Energy Information Administration's (EIA) "U.S. Underground Storage of Natural Gas in 1997: Existing and Proposed"

3. NRG Associates database has been utilized to provide initial estimates of rock and fluid properties data such as porosity, permeability, initial fluid saturation etc. However, these are updated in the

model to ensure consistency with reported AGA Original Gas in Place (OGIP) and maximum working gas capacity.

For potential storage sites (that could be brought on line under favorable economic conditions) we have used depleted oil and gas reservoir data from NRG Associates database. We have used appropriate selection criteria to select depleted oil and gas reservoirs suitable for converting into gas storage sites.

IV. CHRONOLOGICAL HISTORY OF GSAM DEVELOPMENT

IV.1 OVERVIEW OF GSAM DEVELOPMENT HISTORY

1992	***Contract awarded***
1993	***Prototype developed***
1994	***Initial design completed***
1994	***Environmental module initiated***
1994	***Data development task initiated***
1994	***First peer-review meeting***
1995	***Enhancements based on first peer review***
1996	***Utilized for first DOE metrics/planning***
1996	***Improvements to models and data implemented***
1997	***Second peer-review meeting***
1998	***Enhancements based on second peer review***
1998	***Utilized for second DOE metrics/planning***
1999	***Utilized for NPC environmental study***
2000	***Utilized for DOE federal access study***
2001	***Currently used for fourth DOE metrics/planning***

IV. 2 FIRST PEER-REVIEW MEETING – OCTOBER 1994

The first version of GSAM was completed in mid 1994. For the purpose of assuring that GSAM meets DOE objectives, a first peer-review meeting was conducted on October 11-12, 1994 in Pittsburgh, PA. Several industry and government experts were invited to review and identify key areas of improvement, enhancement, modification, and applications in GSAM. The meeting was divided into three sub-groups to review and discuss the following three topics:

- Resource Characterization

- Production and Exploration Model Development, and
- Downstream/Demand Modeling.

The Resource Characterization sub-group recognized the lack of current reservoir pressure data as the most important parameter that could effect the results significantly. Availability of current reservoir pressure data would improve the history matching procedure in GSAM and potentially could change the pressure-production response from the reservoir performance module.

Comparison of GSAM results with the publicly available data such as the Annual Energy Outlook (AEO) was a key recommendation from the Production and Exploration Model Development sub-group. The sub-group also recommended development of environmental sub-module that could feed data directly to the Exploration and Production Module.

Recommendations from the Downstream/Demand Modeling sub-group include modification from two-season to multi-season model and enhancement in gas storage aggregation from regional level to reservoir or active site level. Other suggestions from the review committee were to incorporate better reporting algorithms and to provide detailed documentation for all GSAM modules.

IV. 3 Second Peer-Review Meeting – February 1997

The second GSAM peer-review meeting was conducted on February 5-6, 1997 in ICF offices in Fairfax, VA. The meeting was divided into three sub-groups to review and discuss three different topics as follows:

- Reservoir Data and Modeling (Gas Production and Storage)
- Exploration and Production
- Transportation/Storage/Demand Modeling

After reviewing the overall reservoir data and modeling approaches that GSAM undertakes, the Reservoir Data and Modeling sub-group recognized the lack of current reservoir pressure data as the most important parameter (this was also the main issue in the first peer-review meting). The second most important concern that the review committee had was the lack of a comprehensive storage and production gas reservoir database for Appalachia. It was recommended to investigate the Appalachian Atlas effort at DOE that could provide key insights into the reservoir data for this basin. This would significantly improve the validity of GSAM in Appalachian states such as Pennsylvania, Ohio, West Virginia, New

York, and Kentucky. Capturing reserve growth and documenting the input parameters used in GSAM were also seemed to be very important. It was recommended to provide better documentation on GSAM that could help DOE managers in understanding the source and quality of costing and technology data used in the model. Further, an improved windows-based interface would make GSAM available and useable not only to the engineers but also to the group of users who are not necessarily concerned with the detailed engineering calculations in the routines. This would help the policy makers analyze various scenarios quickly.

The Exploration and Production subgroup recognized the key lack of integration between activities directed at finding, developing, producing, and marketing natural gas and related industry efforts to produce oil. The committee felt that the separation of the activities between gas and oil was inconsistent with the way operators approach investment decisions. Further, the key assumptions necessary to split the industry, including segmenting of the rig fleet between natural gas and oil, and lack of allocation of capital and other constraints across all industry activities could skew GSAM results. The subgroup suggested that, at a minimum, the module should be linked for oil and gas analysis as related to rig utilization, capital spending, and assessment of regional associated-dissolved gas production. Activity by industry, and the resulting production, in these key areas cannot be properly evaluated in a gas-only context.

The Transportation/Storage/Demand Modeling sub-group felt that the recommendations to the Storage Module were critical. This opinion was expressed since the associated changes were relatively easy to incorporate and the benefits would be great in terms of much more realistic storage modeling. Along the same lines, it was believed that the use of four seasons (instead of the current two) would be a great help in making the model more realistic in terms of the economics of storage (as well as other areas). At present, with only two seasons, pipeline usually wins out. It was felt that the extra benefits of four seasons would outweigh the increased complexity/running time in the model. The lack of a more realistic demand side was also a big concern. Two other recommendations: taking into account appliance stock turnovers and inter-fuel competition, addressed this point and were deemed high priority items. Of lesser importance although still of value, was the need to expand the sectors (add transportation), the number of regions, and allow for possible importing and exporting of electricity between demand regions in conformity with reality. It was also mentioned that incorporation of the demand modules from NEMS, which do provide greater detail, would be a worthwhile endeavor. Lastly, other items of great significance concerned the assumption that the linear program (LP) was operating under “perfect foresight”. That is to say, the optimization was performed for all regions and for all years simultaneously. It was believed that running the LP for all regions but one year at a time would speed up the calculations as well as offer a potentially more realistic foresight. The decisions about pipelines, etc. from one year would be carried forward to the next year in this revised approach. In general, speeding up the model was considered important and ways should be examined to achieve this end. Lastly, the incorporation of regional supply curves directly into the integrating LP was believed to be an improvement over the current approach of using a national supply curve.

IV. 4 ANNUAL UPDATES

Annual progresses in the GSAM development project were reviewed, summarized, and documented annually in the form of technical reports. Annual reports were sent to DOE on yearly basis from the start of the project. The annual technical report provides an overview of GSAM methodology, modeling design and development, work plan, and accomplishments of the project during the reporting year. The report also provides recommendations and future work plan to be implemented in the next GSAM contract year. In the following sections we will summarize the recent annual reports (starting from year 1995) which will give an overview of various functionality added over time to GSAM.

CONTRACT YEAR THREE (1995)

The initial phase of GSAM development was successfully completed in mid 1994. Individual modules of the system and critical links between components were fully developed. Data for testing the procedures and processes of all segments of the model were made available for use in future research. At this stage, the model was ready for testing, calibration, and validation to enhance and expand the overall GSAM capabilities. The expected future work included verification of all economic and technology parameters used in the model.

In 1995, various possibilities of GSAM applications were identified. One of them was the analysis of R&D program objects, which was critical to DOE's efforts to focusing its program on the critical needs facing the natural gas industry. Area of improvements was identified and recommended for implementation in GSAM for addressing program and policy questions.

CONTRACT YEAR FOUR (1996)

During the 1995-1996 period, the GSAM project was concentrated in the implementation of recommendations from the first peer-review meeting. Research was focussed on the design work on critical data and model segments required to fully evaluate future market conditions. Development, testing, and integration of an environmental analysis module, including data development, analytical methodology, and full integration into GSAM were in the process of being completed in coordination with DOE. Reservoir level database was finalized. Data development was completed and resource segments were implemented. The highlight of the 1995-1996 period was both the development and implementation of a Canadian resource database and the development of metrics analysis in support of DOE.

Model development task was partly finished. The task was the development and testing of the individual modules that constitute GSAM. Work on this task began with the development and full validation of all individual modules. Model development was complete and much of the essential testing of the modules had been awaiting the availability of resource data. Full integration of all modules was completed and all interfaces, inputs, and outputs were continuing to be tested. The full GSAM system was installed at DOE's Morgantown office and DOE headquarters in Washington DC. Documentation on model development and the users' guide were provided.

Environmental Module (EM) was under development. The objective of design and development of EM was to provide the capability to characterize the impact of changes in environmental regulations and advancements in waste control/mitigation technologies on industry operations, total gas reserves, industry employment, public sector revenues, and, where sufficient data exist, on the environment. This capability will assist DOE in many upcoming analyses of incremental compliance cost impacts, technology evaluations and benefits assessments of different potential future regulatory scenarios.

Data sources to account for all resource types were identified, analyzed, and incorporated for use in GSAM. Methods were developed to analyze, convert, and assign default key data elements for analysis in the type curve and economics models of GSAM.

Sources of updated data were identified, collected, and incorporated into GSAM. Minerals Management Service (MMS) 1996 assessment was scheduled to be incorporated into GSAM data system to provide new estimates for federal offshore resources.

Canadian reservoirs were anticipated to contribute ten percent or more of the overall gas demand in the U.S. over the next several decades. Descriptions of this important resource through data collected by NRG Associates enhanced GSAM database. The full description and consistent analysis of Canadian gas reservoirs substantially enhanced GSAM's capability to assess R&D impacts throughout the North American gas market.

Overall, the data development effort had been successful in utilizing diverse information sources to describe gas reservoirs nationwide. The expansion of data for Canada and offshore, as well as the updating of the undiscovered and known reservoirs throughout the country had substantially improved GSAM credibility.

The first DOE metrics/planning study was completed. The objective of this DOE's natural gas R&D program was to assure that the U.S. gas resource base was capable of meeting the nation's need for low-cost supplies well into the 21st century. The program was directed at providing improved characterization of the nation's gas resource base; improved low-cost technologies for finding, developing, and producing the resource; improved reliability and cost of storage of natural gas; and improved technologies and processes for upgrading the nation's gas supplies to meet customer requirements. The evaluation of the natural gas R&D program indicated the significant benefits accruing to the nation as a result of its activities.

Sensitivity analysis study was completed. The goal of the analysis was to demonstrate the relative sensitivity of GSAM to changes in various model inputs. Runs were completed for the horizontal well and system pressure analyses. The results indicated that attainable changes in technology performance could substantially increase gas production and associated benefits. Based on the GSAM results, the relative impacts of various changes in the input to the model were better understood. This should aid in evaluating the probable impacts of the DOE Natural Gas R&D program.

Summary of key activities completed during the 1995-1996 period include:

- Licensed and screened updated NRG Associates' Significant Oil and Gas Fields of the United States reservoir database as well as new NRG Associates' The Significant Oil and Gas Pools of Canada

- Finalized and tested new databases for U.S. onshore, Appalachia, Canadian conventional reservoirs, Canadian coal and tight reservoirs
- Finalized database structures for use in GSAM and linkage to other systems
- Calibrated the exploration module
- Updated databases for transportation and downstream/demand models
- Developed metrics analysis in support of DOE
- Conducted other briefings for METC and DOE/HQ managers
- Delivered updated User's Guide, models, and databases for METC use and testing
- Conducted initial training on modeling methodology at METC and HQ

Work planned for the next contract year was expected to result in an enhancement of the system, thorough testing, validating, and calibrating. Research activities planned for the next year include:

- Identify, collect, and verify most recent data
- Update GSAM Model
- Implement System Integration/Enhancements
- Environmental Module
- Peer Review
- Policy Evaluation
- Final Research Report

CONTRACT YEAR FIVE (1997)

A major event in GSAM contract year 1997 was the Peer Review Workshop, which provided GSAM with direction and increased focus. The reviewers who participated in the two-day workshop recommended several improvements. The recommendations covered an array of model data and methodology, including improvements to reservoir data and to the E&P, Demand, Integrating, and Storage Modules. Guidance on the design and implementation of the Environmental Module was also given. Like the Environmental Module, the Storage Module also received the attention of the Peer Review Workshop. The Storage Module was completed in 1996, and ready for modifications to meet the specifications recommended by the workshop participants.

Overall GSAM testing was performed in year 1996. In the effort to ensure the validity and long-term reliability of GSAM, the Reservoir Performance Module, the E&P Module, the Demand and

Integrating Modules, and the Storage Module were all tested. The results of the tests were consistent with geologic and financial evidence, and provided conclusions that were intuitive and well-documented.

The upstream and downstream data were revised, including reservoir data, drilling and completion costs, O&M costs, and demand data. GSAM was used to study the effects on the public and private sectors of a royalty relief tax credit for marginal wells. The study was ongoing, and in the past year it produced results which demonstrate GSAM's applicability to a variety of scenarios.

The GSAM run time, both in an integrated fashion as well as for the E&P Module by itself had been greatly reduced. This was done through a combination of hardware improvements and streamlining the logic of the programs involved.

The development of windows version of GSAM (WGSAM) under Microsoft Windows environment (Windows 3.1, Windows 95, and Windows NT) was initiated. The work plan was to develop both front-end Windows interface – to facilitate the input and running of GSAM, as well as back-end Windows interface – to assist in analysis of the output.

Summary of key activities completed during the 1996-1997 period include:

- Analyzed the effects of a royalty relief policy for marginal wells
- Studied the impacts of advanced gas processing technology on natural gas supplies and updated the gas processing module in GSAM
- Modified GSAM to gain an understanding of the market for natural gas storage in order to provide for rigorous evaluation of federal R&D opportunities in storage technologies; we also added a new Storage Reservoir Performance Module and made suitable changes to the integrating linear program. This allows GSAM to thoroughly assess impacts of technology on storage utilization and the market impacts of storage on future gas prices which could in return affect E&P technology application
- Updated upstream and downstream databases
- Enhanced and updated the resource databases for U.S. and Canada for both discovered and undiscovered resources
- Tested and validated GSAM's inputs and outputs to verify their soundness

- Conducted two Peer Review Workshops (one devoted to performance and economic model and data issues, the other focusing on environmental matters) collecting comments and suggestions from a panel of experts from government, industry, and academia
- Decreased GSAM run time by a combination of software and hardware changes
- Performed initial development of a GSAM Windows interface

CONTRACT YEAR SIX (1998)

During the period of performance, July 1, 1997 – June 30, 1998, some at DOE raised questions about the natural gas production profile produced by GSAM. In particular, concern was focused on the predicted decline in gas production after the year 2005. Various working groups were formed to examine specific parts of the model and explore what factors were causing this decline in production. Technology penetration, horizontal wells, as well as resource issues were investigated. Through the investigation of horizontal wells, technology penetration curves, and the GSAM resource base, ICF and DOE were able to better understand and document some of the critical features of the model. These discussions led to a resolution of the “hump” issue, the predicted decline in the natural gas production profile after the year 2005. A thorough review of model structure, including some slight modifications to the Resource Module, and the incorporation of the concept of “reserve growth” in the supply model led ICF to the solution. The incorporation of reserve growth was the primary factor in the solution to the “hump” issue. This concept allowed the GSAM resource base grow to grow in a fashion consistent with both the theory of reserve growth, and the continued increasing production of U.S. natural gas. Reserve growth was found to be a critical element in the dynamics of the North American natural gas market. Having this concept incorporated into GSAM increased the ability of the model to make accurate predictions consistent with the trends in historical reserves data

A methodology, to treat reservoirs that are on Federal lands, was developed in GSAM. This new model was able to address factors affecting the extraction of gas from Federal lands. The Federal policy constraints, the competing desires of preservation and revenue, and other elements, which lead to a development environment different from that on non-Federal lands were modeled in GSAM through a separate technology penetration curve for resource existing on Federal lands. Moreover, the new model was able to study the impact of acceleration in technology penetration in reservoirs on Federal lands due to R&D programs and initiatives of Federal Government agencies, such as DOE. To arrive at a technology penetration rate on Federal lands, two steps were undertaken. The first involved assembling the location and reservoir property data (identifying which reservoirs were on Federally-owned land), and the second was the implementation of a Federal lands technology penetration curve into GSAM.

In accordance with the recommendations of the second peer-review workshop, GSAM's treatment of storage reservoir performance was expanded from the 2-season model to a 4-season model. This expansion led to the incorporation of a 4-season model in the Demand Module. In addition, commensurate with the recommendations of the Peer Review, the decline rate in storage reservoir extraction/injection rates was changed from 5% per year to 3% per year.

Environmental Module, a suite of regulatory models, was under development. This module was developed to add the capability to (1) estimate the impacts of various environmental initiatives on the natural gas E&P sector and (2) provide detailed analyses of the costs and benefits of proposed or anticipated regulations that might have significant effects on the gas E&P industry. These models were intended to support the DOE Office of Fossil Energy's mission of maximizing the recovery of U.S. oil and gas resources through research and development and by working to reduce the costs of effective environmental protection.

The previous Storage Module in GSAM had an incomplete number of potential natural gas storage reservoirs from the Appalachia region. In order to enhance the model, an Appalachian potential storage reservoir database was developed and incorporated into the existing Storage Module. ICF constructed the database by selecting possible storage candidates from existing natural gas producing reservoirs. These reservoirs were added to the database as potential storage reservoirs.

Version 1.0 of Windows GSAM (WGSAM), a fully functional tool to aid in the setup and analysis of GSAM runs, was completed and installed at FETC in Morgantown in the winter of 1997-1998.

Summary of key activities completed during the 1997-1998 period:

- Performed a thorough review of model structure, resolved the "hump" in the production curve
- Implemented the concept of reserve growth in the upstream Exploration and Production Module
- Defined Federal lands in the GSAM database and added a Federal technology penetration curve to analyze Federal lands policy issues
- Developed a 4-season storage routine, and added a 4-season model to the Demand Module

- Performed the programming and initial testing of an annual model
- Continued development of GSAM's separate Environmental Module
- Completed development of a GSAM Windows interface
- Updated the GSAM User's Guide
- Refined GSAM database with respect to Appalachian storage reservoirs and impurity information

CONTRACT YEAR SEVEN (1999)

During the period of July 1, 1998 to September 30, 1999, several enhancements were implemented to GSAM's database and computer model. Programmer's guides for four GSAM major modules were completed. Furthermore, new specification files were created and new modeling approaches were implemented. These modifications and new developments in GSAM improved its overall performance and increased its ability and flexibility to control various modeling parameters.

Programmer's guides for GSAM main modules were produced to provide detailed descriptions of all major subroutines and main variables of the computer code. General logical flowcharts of the subroutines were presented in the guides to provide overall picture of interactions between the subroutines. A standard structure of routine explanation was applied in every programmer's guide. In some of the guides, interactions between the routine itself and its parent and child routines were presented in the form of graphical flowchart. The explanation was presented in the form of step by step description of computer code in the subroutine. The name and release date of the four programmer's guide were:

- Programmer's Guide for Exploration and Production (E&P) Module, January 1999
- Programmer's Guide for Demand and Integrating (D&I) Module, February 1999
- Programmer's Guide for Reservoir Performance (RP) Module, March 1999
- Programmer's Guide for Storage Reservoir Performance Module (SRPM), June 1999

Federal land leasing/development modeling was completed. Modifications in database and computer code were exercised to incorporate Federal land leasing and development modeling in GSAM. The Resource Module was updated, with a logic based on average recoverable reserve fraction of Federal land in the play, to split GSAM undiscovered database into Federal and Private databases. A new specification file (RESAV.SPC) was added to the E&P module to control undiscovered reserve availability in relation with effective penetration rates of exploration drilling. The file stored regional reserve availability percentage of each resource type for Federal and Private lands as a function of time. For the purpose of controlling penetration rates of current and advanced technologies for development and exploration drilling in Federal lands, two new specification files, DTEC_FED.SPC and ETEC_FED.SPC, were added to the E&P module. The specification files stored current and advanced technology incremental penetration rates as a function of time for development and exploration drilling programs. Several changes were implemented in the modeling side of GSAM to incorporate the changes in GSAM database and data specification, and to enhance development and exploration logic for Federal and Private lands. Code modification for the Reservoir Performance (RP) module was minimal. Several minor alterations were performed which include read/write formatting modifications to subroutines for reading the GSAM database, reading play definition file, and writing RP outputs. Major code modification was implemented in the Exploration and Production module. Several modifications to development and exploration algorithms were implemented on top of the basic changes as applied in the RP module. The calculation of undiscovered resource availability was modified to incorporate reserve availability rates specified in new specification file RESAV.SPC. The reserve availability rates were utilized in E&P module as multipliers to the existing exploration technology penetration rate. Product of exploration technology penetration rate and reserve availability rate was used to control the availability of Federal/Private undiscovered resource. Similar to the RP module, only minor changes were made to the Production and Accounting module. The same concept as in the RP module was applied in modifying several subroutines in the PA module. Implementation of the Federal land leasing and development modeling in GSAM enabled it to provide a very precise look at the impact of changing Federal policies on the oil and gas industry.

Offshore database and drilling cost modeling updates were completed. GSAM definition of Gulf of Mexico regions was updated and made consistent with MMS description of western, central, and eastern Gulf of Mexico areas. GSAM offshore database was modified to include undiscovered plays in the eastern Gulf of Mexico and Atlantic Offshore regions. Offshore drilling cost formulation was developed and implemented in GSAM modules. Modifications in offshore database and drilling cost modeling maintained the consistency of GSAM database and contributed to more accurate GSAM predictions.

Tight reservoir type-curve modeling update was completed. The hydraulically fractured well and horizontal well models for tight reservoirs in the RP module were modified. The updated RP module eliminated the doubling effect (production from horizontal wells twice the production from hydraulically fractured wells) in the previous version of the RP module. The fractured well model was verified with the conventional model. The validation runs showed that the fractured well model collapsed to the conventional well model for small fracture half-lengths.

Storage Reservoir Performance Module (SRPM) was updated and released. The new SRPM model utilized the reservoir level properties from 1997 releases of American Gas Association (AGA) and Energy Information Administration (EIA) to determine the characteristics of underground gas storage in the United States. Altogether, 100 new storage reservoirs were added to the SRPM database. These new additions were generated based on differences in number of reservoirs reported in the AGA and the EIA. Some modeling aspects were modified to provide the SRPM with better procedure for reservoir property adjustment, more flexibility in time step sizes, and consistent methodology in Absolute Open Flow Potential (AOFP) calculations.

A new exploration-drilling algorithm was developed and successfully implemented in the EP module. In the previous EP module, each successful exploration drilling effort was assumed to find three accumulations (in one undiscovered field): one accumulation in the current field size class (FSC) and two accumulations in smaller FSCs. This assumption was found to be optimistic and causing number of exploration wells to be lower than expected. The very first attempt to solve the exploration drilling issue was to redefine the assumption of successful exploration drilling utilized in the E&P module. In the new algorithm, one successful exploration drilling effort was assumed to find only one accumulation or for a success rate of 100%, each accumulation explored represented by one exploration well. The new exploration drilling assumption was implemented in the E&P module. The new GSAM model (with new successful exploration drilling assumption) improved the number of exploration well prediction significantly.

Selection criteria for exploration and development project in the EP module was modified. The project selection criterion in the previous EP module ranked the exploration and development projects on the basis of minimum acceptable supply price (MASP). The project with lowest MASP was assumed to be the most profitable. However, it was realized that in many cases the assumption of lower MASP resulting in greater profitability was not necessarily true. The MASP calculation did not take into account the effect of drilling rig availability or capacity in the region where the project was located. Therefore, a project with low MASP but located in a region with shortage in drilling rig capacity should not be given a high rank unless the project was still economic by adding cost associated in transporting rig capacity from another region. Consistent breakeven drilling cost factor (BDCF) formulation was developed and implemented in the EP module. The projects were ranked based on both MASP and BDCF. The BDCF was instrumental in controlling utilization and movement of the regional rig capacities especially when there were shortages in regional rig capacities.

Regional reserve growth function was updated and the EP module was recalibrated. In the past, the regional reserve growth function of the E&P module was not completely accurate because of insufficient data. New reserve growth data from the USGS was cross-mapped onto GSAM regions and the annual growth rate was placed into the specification file RESAVRG.SPC for both specific regions and the United States as a whole. The updated values were the exact projections of the USGS for the regional reserve growth rate through the year of 2020.

Summary of key activities completed during the 1998-1999 period:

- Produced programmer's guides for Reservoir Performance Module, Storage Reservoir Performance Module, Exploration and Production Module, and Demand and Integrating Module
- Designed and implemented Federal land leasing/development model into GSAM
- Updated offshore database to include Eastern Gulf of Mexico and Atlantic Offshore undiscovered fields
- Developed and implemented water-depth specific drilling cost model for offshore wells
- Enhanced tight reservoir model to improve deliverability calculations from hydraulically fractured reservoirs and incorporated consistency with horizontal well computations
- Updated Storage Reservoir Performance Module (SRPM) database consistent with published data from American Gas Association (AGA) and Energy Information Administration (EIA)
- Redesigned numerical model of the SRPM to produce consistent data entry of injection/extraction program for the Annual Demand and Integrating Module
- Modified exploration drilling algorithm to improve accuracy of GSAM predictions for exploration wells drilled
- Modified breakeven drilling cost formulation in project selection criteria to incorporate selection based on profitability and not production
- Implemented USGS reserve growth function into Exploration and Production Module
- Implemented issue-specific environmental cost model into Exploration and Production Module

- Updated database and mathematical model of Industrial Demand Module to account detailed information on boilers, cogeneration/nonutility generation, process heat, and feedstock
- Updated GSAM annual model to take into account variation of wholesale-to-retail markups with respect to time, weather influence, and heat rate variation by vintage
- Modified cost file in Production and Accounting Module to account for regional cost variation consistent with the cost files in Reservoir Performance Module

CONTRACT YEAR EIGHT (2000)

During the period of October 1999 to March 2000, several enhancements were implemented and proposed to GSAM's database and computer model. These modifications and new developments in GSAM improved its overall performance and increased its ability. Four potential modeling enhancements were proposed for implementation in GSAM. The proposed tasks were found to be critical for improving the accuracy of GSAM predictions.

RESERVE BOOKING (PROPOSED)

Despite the fact that it impacts most of the stakeholders in the oil and natural gas industry, including field operators, royalty owners, Government, and the investing community, reporting of reserves (booking, as it is commonly referred to) is one issue that is not practiced in a consistent manner. It is an issue that also involves various parties within an organization – from geologists to accountants. The absence of a well-defined set of guidelines or "best practice" approach to booking reserves, and the fact that oil and natural gas resource development is subject to significant geologic and market uncertainties, further contribute to the prevailing confusion.

Although a lot has been accomplished by the Society of Petroleum Engineers/World Petroleum Congress joint forum in addressing the definitional issues, the specific issue of the volumes "booked" in specific circumstances has not been addressed. There is gross inconsistency in the interpretations of guidelines published by various regulatory bodies. Quantifying geologic success factors is not an exact mathematical formulation, and the industry most often resorts to reporting a range of possibilities when it comes to determining the potential size of a hydrocarbon find. From an accounting perspective, the investing communities and in particular the Securities and Exchange Commission (SEC) and Financial Accounting Standards Board (FASB) have recognized the importance of having a set of guidelines that can be applied across-the-board to all the oil and natural gas operating companies. However, even the SEC and FASB statements on this issue are fairly vague, leaving much to individual interpretation of the rules.

Our inquiries to some of the major operating companies has confirmed that a wide variation exists between the different operating companies, much of it reflects the financial position and relative size of the company in the industry. Major oil companies typically tend to be conservative as this helps smooth business cycles. To the contrary, small independent producers tend to be aggressive as it has a direct bearing on their ability to generate project financing from the capital markets. Most other companies fall somewhere within this spectrum depending on their own perception of their relative financial strengths and risks.

We have identified a simple mathematical approach to improving the representation of “reserves booking” procedures which can be used as the “guiding principle”. It is currently not modeled in GSAM. This approach, when implemented in Gas Systems Analysis Model (GSAM), could significantly enhance the way “reserves addition” is computed, and will eventually impact the calculations of finding and development (F&D) costs.

The objectives of the study were threefold:

- To examine the driving forces behind the conventions and norms followed by the various industry stakeholders for reporting and booking reserves of oil and gas relative to the developmental activities followed by the operators. This step includes developing an understanding of the basic terminology, the standardized set of guidelines established by Federal government and other regulatory bodies, such as the SEC and FASB.
- To conduct an examination of historical reserves data available in the NRG Associates’ 1999 version of “Significant Oil and Gas Fields of U.S. Database”. This will provide a preliminary insight into reserves booking over time.
- To propose and examine a simple mathematical model that can serve as the guiding principle or a preliminary rule-of-thumb for booking natural gas reserves.

STACKED RESERVOIRS/GEOLOGICAL PLAYS (PARTIALLY IMPLEMENTED AND PROPOSED)

Natural gas and petroleum bearing reservoir rock formations are created in various aquatic depositional environments – differing in degree, scale, and age. Some environments, particularly marine, tend to deposit sediments continuously. This results in what typically characterizes the sedimentary rocks – layers of depositions. With formation and migration of hydrocarbons to these formations, it is therefore very likely for the natural gas and petroleum to be located in more than one of these layers and thus result in there being multiple reservoirs stacked on top of each other.

Presence of stacked reservoirs in a single field area poses a serious challenge for the field operator. In a field with a single reservoir, the decision is driven primarily by the economics of developing a set of wells to drain that one reservoir. In contrast, operators in a multi-reservoir situation are faced with other options – do they produce the reservoirs sequentially, simultaneously, or through separate development efforts? Geology, engineering, technology, and standard operating practices in addition to economics drive the answer to this dilemma.

This has direct implications on the level of production that can be accomplished from a field, a basin, and a region. In modeling production from future discoveries from the undiscovered reservoirs, it is important to represent this appropriately. Currently, the Gas Systems Analysis Model (GSAM) has no way of treating stacked reservoirs as one unit for onshore resource, and therefore, may not apply realistic development strategies in all cases. However, the error introduced in the production forecast from the model is only as significant as the extent of "stackedness" in the undiscovered basins in the GSAM database.

The primary objective of this study was to conduct an examination of historical data available in the NRG Associates' 1999 version of "Significant Oil and Gas Fields of U.S. Database". This effort provides preliminary insight on the relative contribution from stacked reservoirs to current field production in the United States and also helps identifying the regions likely to contain high concentrations of undiscovered stacked reservoirs. The study also estimates the relative importance of stacked reservoirs to future production, and proposes a series of steps on how to implement the changes to the Gas Systems Analysis Model (GSAM).

Our analysis indicates some interesting facts. While the number of producing fields (oil and gas) containing multiple stacked reservoirs was around 25% of all the fields, the total reserve endowment in these fields represent more than 67% of the total resource base. In the case of oil fields, 79% of the total proved reserves remains in stacked reservoirs, and in the case of gas fields, nearly 54% of the total proved reserves remains in stacked reservoirs.

Investigation of undiscovered resource data indicated that Texas and Louisiana have the highest potential for stacked reservoirs and have high discovery potential. Most of the geologic plays in these states are a continuation of the Gulf of Mexico plays which are deposited under marine environments and are highly stacked.

Minor modifications in the modeling structure were implemented in GSAM to capture reservoir stacking for the Gulf of Mexico region. This simple modification (only to Gulf of Mexico producing reservoirs) was incorporated in the GSAM 2000 version delivered to NETL. The 14th update of NRG Associates data indicated that there are quite a few fields showing high level of reservoir stacking (some fields with more than 10 reservoirs). The discovered reservoir database for Gulf of Mexico-Central and Gulf of Mexico-West regions were modified to treat the stacked reservoirs as a common field. The properties of the individual reservoirs were appropriately aggregated to represent one unified field. This improved GSAM characterization of producing reservoirs in the Gulf of Mexico region. This helped in modeling the existing development practices in the Gulf of Mexico and the resulting economics more appropriately.

It is recommended that pilot test runs be conducted with GSAM for regions with the highest potential for stacked reservoirs, and where enough natural gas resources are present to justify modifications in the model. For these regions, stacking algorithm for reserve development decisions will need to be implemented. The results should be evaluated to determine the benefits of making a large-scale change to the GSAM modeling framework.

The objectives of the study were as follows:

- To conduct an examination of historical data available in the NRG Associates' 1999 version of "Significant Oil and Gas Fields of U.S. Database". This will provide preliminary insight on the relative contribution of stacked reservoirs to current field production in the United States and identify the regions likely to contain high concentrations of undiscovered stacked reservoirs.
- To estimate the relative importance of stacked reservoirs to future production.
- To propose a series of steps on how to implement the changes to the model.

TIGHT GAS RESOURCE UPDATE (PARTIALLY IMPLEMENTED AND PROPOSED)

The GSAM Resource Module provides reservoir-specific information to the GSAM Reservoir Performance Module. For the undiscovered gas resource, the GSAM Resource Module creates a database of reservoir properties based upon the geological plays defined by the US Geological Survey (USGS). The reservoir level data (such as porosity, permeability, pay zone thickness, water saturation, etc.) is provided by the NRG Associates reservoir database (NRG's 14th update released in 1999), USGS (US Geological Survey, Digital Data Series 30, Release 2, 1996), and Gas Research Institute (GRI). Within each USGS play, values of key reservoir properties for the play are assigned to undiscovered fields of various sizes. The undiscovered fields are differentiated by reservoir characteristics such as porosity, permeability, and vertical thickness of the pay zone.

To effectively model exploration and development timing and efficiency, the various field size classes within plays should be differentiated as much as possible. The distribution of reservoir characteristics across different field size classes determines the cost structure of the undiscovered resource in the play. If most undiscovered field size classes have similar cost structures, the undiscovered resource exhibits poor sensitivity to changes in exploration, development, environmental and other costs. Thus, for a given level of cost impacts on the undiscovered resource, almost the entire resource base tends to be either economic or uneconomic to find and develop.

The purpose of this study was to revise the characterization of the undiscovered tight gas resource in DOE's Gas System Analysis Model (GSAM). The objectives of the assignment were two-fold:

- To update the reservoir characterization of the undiscovered tight gas resource in the western United States by updating the reservoir characteristics of each play with relevant reservoir properties obtained from current reservoir engineering and geologic literature.
- To improve the differentiation among undiscovered tight gas plays and among field size classes within individual play. Differentiation of the tight gas resource was accomplished by distributing reservoir properties such as porosity, in situ permeability, water saturation, and pay thickness across field size classes to reflect the reservoir characteristics reported in the geologic and engineering literature.

The focus of this effort was the undiscovered tight gas resource in the Rockies Foreland, San Juan Basin, and Williston Basin GSAM regions. A future objective of this work is to expand the improved characterization of the undiscovered tight gas resource to other GSAM regions, particularly the Appalachian and the Gulf Coast regions. The latest version of GSAM “GSAM 2000” does incorporate the tight gas resource enhancements described in this paper. This version of GSAM was delivered to NETL in March 2000.

INTER-FUEL COMPETITION (PARTIALLY IMPLEMENTED AND PROPOSED

The purpose of the project was to study the statistical relationship between inter-fuel competition (including electricity), population, gross state product (GSP), and weather on natural gas demand in both the residential and commercial sectors. The project was to conduct several regression analyses by census region to evaluate the possible relationships between these variables (i.e. commercial and residential fuel demand and prices, GSP, and weather). The project consisted of four basic components:

- Gather the data from a variety of publicly available sources
- Format the data to be read by the SAS statistical programs
- Write the SAS programs to perform regression analyses of the data
- Analyze the model results

At the current time, the first two components have been completed. Due to the lack of additional funding required to complete the remaining points, these items have not been completed. We anticipate successfully completing them pending additional funding, based on our experience with a similar study concerning inter-fuel competition in the industrial demand sector, that was undertaken and successfully completed last year.

V. BIBLIOGRAPHY

Anderson, Kenneth E. and Bill D. Berger. Modern Petroleum. Tulsa, OK: The Petroleum Publishing Co., 1978.

Arps, J.J.:"Estimation of Primary Reserves," Trans., AIME, 1956.

Blasingame, T.A. and Poe Jr., B.D.: "Semianalytic Solutions for a Well with a Single Finite-Conductivity Vertical Fracture," Tech. Paper SPE 26424, ATCE, 1993.

Canadian Geologic Survey: "Western Canada Basin Conventional Gas Resource Estimated at 232 TCF," *Oil and Gas Journal*, October 25, 1993, p. 92-94.

Carr, N.L., Kobayashi, R. and Burrows, D.B.: "Viscosity of Hydrocarbon Gases Under Pressure," Trans., AIME, 1954.

Code of Federal Regulations 17 for Commodity and Securities Exchanges – *"A codification of documents of general applicability and future effect as of April 1, 1999"*. 17 CFR Ch. II (210.4-10). Office of the Federal Register.

Colebrook, C.F.: "Turbulent Flow in Pipes, with Particular Reference to the Transition Region Between the Smooth and Rough Pipe Laws," *Jour. Inst. Civil Engr.* (London), 1938.

Craft, B.C. and M.F. Hawkins, *Applied Petroleum Reservoir Engineering*, 1959, p. 42.

Craft, B. C., W. R. Holden, and E. D. Graves Jr. Well Design: Drilling and Production. Englewood Cliffs, NJ: Prentice-Hall Inc., 1962.

Chowdiah, P., 1987, Two -Phase Flow in Tight Gas Sands (MWX data) in Proceedings of Unconventional Gas Recovery Contractors Meeting, July 1987, Morgantown Energy Technology Center.

Cluff, S.G., Cluff, R.M., 1996, Petrophysical analysis of the Frontier Formation, Whiskey Buttes Field, Lincoln Co., WY in Formation Evaluation of Low Permeability Reservoirs in the Rocky Mountain Basins, April 29, 1996, Denver, Colorado.

Crocker, M.A., 1996, Petrophysics of the overpressured tight formations of the greater Green River Basin in Formation Evaluation of Low Permeability Reservoirs in the Rocky Mountain Basins, April 29, 1996, Denver, Colorado.

Davis, John C. Statistics and Data Analysis in Geology. New York: John Wiley and Sons Inc., 1973.

Dutton, S.P., Hamlin, H.S., Laubach, S.E., 1995, Geologic Controls on Reservoir Properties of Low Permeability Sandstone, Frontier Formation, Moxa Arch, Southwestern Wyoming, Gas Research Institute Report 234, 89 pp.

Finley, R. J.,1985, Reservoir properties and gas productivity of the Corcoran and Cozzette tight sandstones, Colorado in Symposium on Low Permeability, SPE/DOE Technical Paper 13852.

Gautier, D.L. and others, eds., 1996, 1995 National Assessment of United States Oil and Gas Resources, U.S. Geological Survey Digital Data Series, DDS-30.

Gas Research Institute, *Gas Research Institute's Baseline Forecast*, 1993-99.

Hall, H.N.: “Compressibility of Reservoir Rocks,” Trans., AIME, 1953.

Hall, K.R. and Yarborough, L.: “A New Equation of State for Z-Factor Calculations,” *Oil and Gas Journal*, 1973.

Hollberg, J., Dahm, E., Bath, J., 1985, Geology and production performance of the Niobrara low-permeability reservoir in the Denver-Julesberg basin in Symposium on Low Permeability Reservoirs, SPE/DOE Technical Paper 13886.

J.G. Ross, *“Booking Reserves”*. Prepared for presentation at 1998 SPE Annual Technical Conference and Exhibition held in New Orleans, September 1998. Society of Petroleum Engineers.

King, G.R.: “State-of-the Art in Modeling of Unconventional Gas Recovery,” Tech. Paper SPE 18947, Presented at the Joint Rocky Mountain Regional/Low Permeability Reservoirs Symposium and Exhibition, 1989.

Krystinik, L.F., 1996,Greater Green River basin production improvement project, status report: Stratos #1 well, core analysis results in Formation Evaluation of Low Permeability Reservoirs in the Rocky Mountain Basins, April 29, 1996, Denver, Colorado.

Kukal, G.C., Simons, K.E., 1985, Log analysis techniques for quantifying the permeability of sub-millidarcy sandstone reservoirs, in Symposium on Low Permeability Reservoirs, SPE/DOE Technical Paper 13880.

Kuuskraa, V.A., 1996, Advanced technologies for producing massively stacked lenticular sands in Formation Evaluation of Low Permeability Reservoirs in the Rocky Mountain Basins, April 29, 1996, Denver CO.

Merewether, E.A., 1987, Plays for oil and gas in the Raton Basin, south-central Colorado and northeastern New Mexico, USGS Open-File Report 87-450A

Mullarkey, J., 1996, Formation evaluation of Upper Cretaceous reservoirs in the Greater Green River basin in Formation Evaluation of Low Permeability Reservoirs in the Rocky Mountain Basins, April 29, 1996, Denver CO.

Moody, L.F.: “Friction Factors for Pipe Flow,” Trans., ASME, 1944.

National Energy Board: <u>Unconnected Gas Supply Study</u>, January 1995.

Natural Gas Potential in Canada, A Report by the Canadian Gas Potential Committee, 1997, Natural Gas: Meeting the Challenges of the Nation's Growing Natural Gas Demand, Volumes I, II, and III, A Report of the National Petroleum Council (NPC), December 1999.

NRG Associates Oil and Gas Database.

Ozkan, E. et al: "Horizontal-Well Pressure Analysis," Trans., SPE 16378, SPEFE, 1989.

Ozkan, E. et al: Supplement to SPE 16378, "Horizontal-Well Pressure Analysis," Tech. Paper SPE 20271, UNS, 1989.

Pollastro, R.M., Scholle, P.A., 1986, Exploration and development of hydrocarbons from low-permeability chalks- an example from the Upper Cretaceous Niobrara Formation, Rocky Mountain region in Spencer, C.W. and Mast, R.F., eds., Geology of Tight Gas Reservoirs, AAPG Studies in Geology #24, pp. 129-141.

Ramsey James B., The Economics of Exploration for Energy Resources, JAI Press, 1981.

Scotia Group, 1995, Reserve Estimates in Western basins Part III: Uinta Basin, Report for DOE, FETC, Contract DE-AC21-91-MC28130.

Spencer, C.W., 1989, Review of characteristics of low permeability gas reservoirs in the western United States, AAPG Bulletin, V. 73, No.5, pp. 613-629.

Slider, H. C. Worldwide Practical Petroleum Reservoir Engineering Methods. Tulsa, OK: PennWell Publishing Co., 1983.

Smith, R.V.: *Practical Natural Gas Engineering*, 1990.

Society of Petroleum Engineers, *Petroleum Engineering Handbook*, Richardson, TX, 1987, p. 24-12.

"SPE/WPC Reserves Definitions Approved", Journal of Petroleum Technology, May 1997.

Stermole, Franklin J., and Stermole, John M., Economic Evaluation and Investment Decision Methods, 8th Edition, Investment Evaluations Corporation, Golden, Colorado, 1993.

Statement No. 69 – "Disclosure about Oil and Gas Producing Activities – an amendment of FASB Statements 19, 25, 33, and 39". Issued 11/1982. Financial Accounting Standards Board.

Standing, M.B. and Katz, D.L.: "Density of Natural Gases," Trans., AIME, 1942.

U.S. Geological Survey, Digital Data Series 30, Release 2, 1996.

van Everdingen, A.F. and Hurst, W.: "Application of the La Place Transformation to Flow Problems in Reservoirs," Trans., AIME, 1949.

Warren, J.E. and Root, P.J.: "The Behavior of Naturally Fractured Reservoirs," *Society of Petroleum Engineers Journal*, 1963.

White, D.A.: "Assessing Oil and Gas Plays in Facies-Cycle Wedges," *The American Association of Petroleum Geologists Bulletin*, v.64, no. 8, 1980.

White, L.P.: "A Play Approach to Hydrocarbon Resource Assessment and Evaluation," *JAI Press*, 1981.

Wichert, E. and Aziz, K.: "Calculate Z's for Sour Gases," *Hydrocarbon Processing*, 1972.

Weimer, R.J., Sonnenberg, S., Young, G., 1986, Wattenberg Field, Denver Basin in Spencer, C.W. and Mast, R.F., eds., Geology of Tight Gas Reservoirs, AAPG Studies in Geology # 24.

Yarborough, L. and Hall, K.R.: "How to Solve Equations of State for Z-Factors," *Oil and Gas Journal*, 1974.

ICF CONSULTING

USER'S GUIDE AND PROGRAM SUMMARY FOR THE GAS SYSTEMS ANALYSIS MODEL (GSAM)

FINAL REPORT

Volume II – User's Guide

For:

U.S. Department of Energy
National Energy Technology Laboratory
Morgantown, West Virginia
Under Contract Number: DE-AC21-92MC28138

By:

ICF Consulting, Inc.
Fairfax, Virginia

February 2001

DISCLAIMER

TABLE OF CONTENTS

Page

TABLE OF CONTENTS (CONTINUED)

TABLE OF CONTENTS (CONTINUED)

TABLE OF CONTENTS (CONTINUED)

LIST OF FIGURES **Page**

LIST OF TABLES **Page**

I. INTRODUCTION

This document provides details on the data, models, required inputs, and output formats for the Gas Systems Analysis Model (GSAM). It was prepared as a guide to the use of GSAM in conducting analyses of the technology and economics of domestic gas production. The report is organized by chapters that correspond to the major functional components of the GSAM system.

The purpose of this document is to provide the information necessary to configure and operate the model for many analytical purposes. Additional details on the scientific and technical parameters, assumptions, and procedures of individual modules and logic for the overall GSAM system are contained in various topical reports prepared under the development contract.

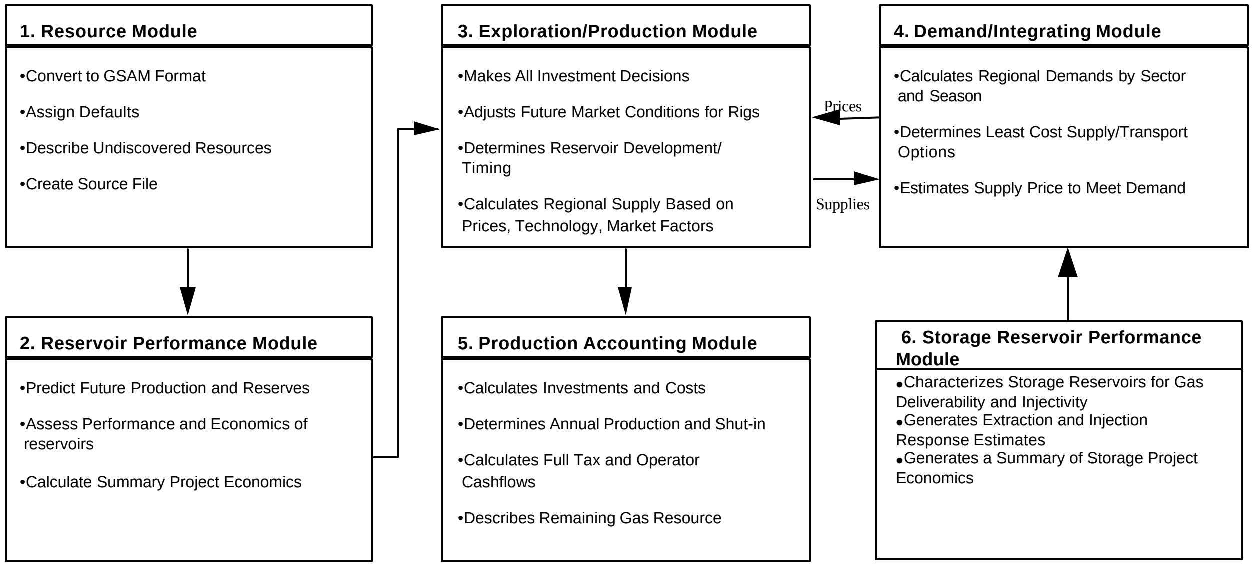
II. DISCUSSION OF MAJOR MODEL COMPONENTS

Figure 1 shows the six key components of GSAM as they relate to and interact with each other. The modular design of GSAM allows analyses to be conducted more efficiently by using intermediate databases and by allowing for the alteration of various assumptions.

The six primary model components include:

- Resource Module
- Reservoir Performance Module
- Exploration and Production Module
- Demand and Integrating Module
- Production Accounting Module
- Storage Reservoir Performance Module

Figure 1
Major Components of GSAM



1. Resource Module
•Convert to GSAM Format
•Assign Defaults
•Describe Undiscovered Resources
•Create Source File
3. Exploration/Production Module
•Makes All Investment Decisions
•Adjusts Future Market Conditions for Rigs
•Determines Reservoir Development/ Timing
•Calculates Regional Supply Based on Prices, Technology, Market Factors
4. Demand/Integrating Module
•Calculates Regional Demands by Sector and Season
•Determines Least Cost Supply/Transport Options
•Estimates Supply Price to Meet Demand
Prices
Supplies
2. Reservoir Performance Module
•Predict Future Production and Reserves
•Assess Performance and Economics of reservoirs
•Calculate Summary Project Economics
5. Production Accounting Module
•Calculates Investments and Costs
•Determines Annual Production and Shut-in
•Calculates Full Tax and Operator Cashflows
•Describes Remaining Gas Resource
6. Storage Reservoir Performance Module
•Characterizes Storage Reservoirs for Gas Deliverability and Injectivity
•Generates Extraction and Injection Response Estimates
•Generates a Summary of Storage Project Economics

The **Resource Module** translates data into the structure required by GSAM. It currently utilizes NRG Associates' (1997 and 1994 versions) databases for the United States (latest production data for year 1997) and Canada (latest production data for year 1994) for discovered producing reservoirs. United States Geologic Survey (USGS, 1995 resource estimates from DDS 30 CD-ROM, Release 2) estimates have been used for the undiscovered onshore gas resource for the United States. Minerals Management Service (MMS) estimates and ICF's internal reports have been used for the undiscovered offshore gas resource for the United States. Finally, Geological Survey of Canada estimates for undiscovered conventional and hypothetical onshore gas resource for Canada, NPC estimates for undiscovered onshore tight gas resource, and Alberta Geological Survey estimates for undiscovered coalbed methane resource for Canada have been used. This Module analyzes the data to ensure the consistency of the input parameters, and it provides a basis for calculating or assigning defaults for any missing data elements which may be required by other modules.

The **Reservoir Performance Module** estimates future production based on unique reservoir properties, technologies, and costs. This Module also performs summary economic analyses to measure investment alternatives in the **Exploration and Production (E&P) Module**. All decisions concerning investment options are simulated in the E&P Module, which uses a defined gas price forecast, or a range of alternative prices, to determine future project attractiveness and timing. The Module's analytical procedures consider the impact of changing prices, market conditions, and technology constraints in determining investment priorities.

If a balanced supply and demand forecast is desired, intermediate results of the E&P module are used in the **Demand and Integrating Module**. This module calculates future demand for gas by region and sector and solves for a balanced equilibrium price forecast, accounting for regional variations in production costs, transportation capacities and costs, and seasonal demand by sector. The final balanced price forecast, or assigned price forecast, is then used in the **Production Accounting Module** to reconstruct the full net cashflow of individual projects selected in the E&P Module for development. It uses economic data input consistent with that in the Reservoir Performance Module. The **Storage Reservoir Performance Module** estimates injectivity, deliverability, levelized investment costs, and variable O&M costs for over 360 active storage reservoirs and over 120 potential storage reservoirs. The response is sent directly to the **Demand and Integrating Module** of GSAM.

A GSAM analysis can involve running all or some of these individual modules. For example, if the resource base being analyzed is not changed, the existing GSAM Reservoir Database can be used without reanalyzing the raw data using the Resource Module. If a gas price forecast has been independently determined, gas exploration and production activities can be evaluated without using the Demand and Integrating Module. The Production Accounting Module is only used once final Exploration and Production Module results are obtained. Similarly, if an analysis only considers changes in gas prices, the Reservoir Performance Module step can be skipped. The following table provides a quick guide to determining which modules would be required to be run based on the desired analysis.

Table 1
Summary of GSAM Modules Used in Various Analyses

	Resource Module	Reservoir Performance Module	Storage Reservoir Performance Module	Demand/ Integrating Module	Production Accounting Module	Exploration/ Production Module
New Resource Description	X	X	X	If resulting Balanced price forecast required	X	X
Change in Costs		X	X	If resulting Balanced price forecast required	X	X
Change in Market Conditions				If resulting Balanced price forecast required	X	X
Change in Gas Price (Prices defined)				N/A	If full revenue expenditure, and tax summary is required	X
Change in Supply/ Demand Factors (solve for gas price)				X	If full revenue expenditure, and tax summary is required	X

The following chapters describe the processes for running each individual module under various assumptions. Each module is described based on the inputs, basic analytical procedures, and resulting outputs. The appendices contain sample input and output files, including the files that are passed between modules and intermediate data files.

III. GSAM File Locations

The entire GSAM model is set up in a single directory, which here is :\GSAM, but may be any name, as no piece of the model is dependant on this main directory's name. The following subdirectory structure is used:

:\GSAM\RESOURCE	Files common to both undiscovered and discovered resource, for Canada and the U.S.
\USDISC	U.S. discovered resource database files and programs
\USUND	U.S. undiscovered resource database files and programs
\CANDISC	Canada discovered resource database files and programs
\CANUND	Canada undiscovered resource database files and programs
:\GSAM\RESVPERF	Executable, batch files, specification files and *.GSM files
\DATA	Data files for the Reservoir Performance Module
\FORT	Source code and common blocks
:\GSAM\EXPLPROD	Executable, batch files, and data files
\FORT	Source code and common blocks
\NEWS	Binary files and specifications for creating them
\TMP	Temporary storage for .DEC,.PRD, .ENV, .GSM files to make binary files and to run horizontal/vertical selection program
\TMP\VERTICAL	Temporary storage for .DEC,.PRD, ASM files, etc. when vertical wells are used in Reservoir Performance Module
\TMP\HORZ	Temporary storage for .DEC,.PRD, ASM files, etc. when horizontal wells are used in Reservoir Performance Module
\ENV	State-specific environmental files created from DOE's Environmental Module
:\GSAM\DEMDINTG	Executable, batch files, and data files
\FORT	Source code and common blocks
:\GSAM\PRODACCT	Executable, batch files, and specification files
\DATA	Data files for the Production Accounting Module (similar to files located in \GSAM\RESVPERF\DATA directory)
\FORT	Source code and common blocks
:\GSAM\SRPM	Executable, batch file, and specification files
\DATA	Data files for the Storage Reservoir Performance Module
\FORT	Source code and common blocks

As indicated by this structure, all modules are executed from the principal subdirectory for the module (e.g., :\GSAM\RESVPERF for the Reservoir Performance Module). Various batch files have been created to rename and copy files, as required, to efficiently execute each module.

Only two files are in the main (e.g., :\GSAM) subdirectory. One is the DOSXMSF.EXE file, which is needed to run some FORTRAN programs and must be in a directory, which is specified in the PATH of the AUTOEXEC.BAT file. Note that this file is necessary to run the Reservoir Performance, the Exploration and Production, and the Production Accounting Modules. The second required file is RUN386.EXE, which is necessary only for the Demand and Integrating Module. Some of the model's main outputs are stored as follows:

:\PRODACCT\NAT.OUT	National Production and Economic Estimates
:\PRODACCT\REGION.OUT	Regional Production and Economic Estimates OR,
:\PRODACCT\STATE.OUT	State Production and Economic Estimates
:\DEMDINTG\GSAMSLN.FLE	Supply/Demand Summary Report, well head prices and end-use prices by Region and sector, etc.
:\DEMDINTG\GSAMSLN.RPT	Detailed Seasonal Report of Supply and Demand including transportation capacities
:\EXPLPROD\PRODSUMM.OUT	Exploration, Production, and Reserves Summary
:\EXPLPROD\SUPPSUMM.OUT	Supply summary by region and resource type

IV. RESOURCE MODULE (1)

A. Summary Description of the Resource Module

The Resource Module converts resource data into reservoir-specific information in a format that the GSAM Reservoir Performance Module can use. The Module has been developed in Statistical Analysis System (SAS) to allow future research to evaluate and conduct statistical sensitivity analyses within GSAM. Although much of the data is from NRG Associates and the USGS, the input data can be taken from any source, provided the data can ultimately be described by individual reservoirs assigned to individual, geologically-based plays. Any new data of this type may be incorporated, as long as the SAS code is altered for the new data. The plays must also be uniquely assigned to individual geographic regions used elsewhere in GSAM. Before running this Module, the user should have the following files, organized by nation (Canada or U.S.), resource type, and file type.

B. United States Resource Module

B.1. Discovered Resource

Input Files

Name	Description	Location
RMASTER.DAT	Reservoir-specific database from NRG Associates	\RESOURCE\USDISC
FMASTER.DAT	Field-specific database from NRG Associates	\RESOURCE\USDISC
FEDGAS.DAT	Federal/Private Land Flag Data from Earth Science Associates (original file)	\RESOURCE\USDISC
FEDGASCH.DAT	Federal/Private Land Flag Data from Earth Science Associates (updates)	\RESOURCE\USDISC
PLY_DFN.SPC	Play-level properties (ICF Consulting)	\RESOURCE\USDISC
GOMWDEP.DAT	Gulf of Mexico Water Depth as a function of geographic play Data from MMS	\RESOURCE\USDISC

Program Files

Name	Description	Location
NRG.SAS	SAS Routine to create GSAM discovered database	\RESOURCE\USDISC

Output Files

Name	Description	Location
GSAM01.GSM	GSAM discovered reservoir database for supply region 1, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM02.GSM	GSAM discovered reservoir database for supply region 2, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM03.GSM	GSAM discovered reservoir database for supply region 3, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM04.GSM	GSAM discovered reservoir database for supply region 4, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM05.GSM	GSAM discovered reservoir database for supply region 5, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM06.GSM	GSAM discovered reservoir database for supply region 6, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM07.GSM	GSAM discovered reservoir database for supply region 7, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM08.GSM	GSAM discovered reservoir database for supply region 8, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM09.GSM	GSAM discovered reservoir database for supply region 9, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM10.GSM	GSAM discovered reservoir database for supply region 10, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM11.GSM	GSAM discovered reservoir database for supply region 11, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM12.GSM	GSAM discovered reservoir database for supply region 12, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM15.GSM	GSAM discovered reservoir database for supply region 15, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM16.GSM	GSAM discovered reservoir database for supply region 16, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM17.GSM	GSAM discovered reservoir database for supply region 17, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM18.GSM	GSAM discovered reservoir database for supply region 18, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM19.GSM	GSAM discovered reservoir database for supply region 19, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
GSAM99.GSM	GSAM additional discovered reservoir database for supply and reserves matching, (full format, 13 lines per reservoir)	\RESOURCE\USDISC
APPL.GSM	GSAM additional discovered reservoir database for the Appalachian region, (1 line per reservoir, data from ICF, state publications and GRI)	\RESOURCE\USDISC

B.2. Undiscovered Resource

Input Files

Name	Description	Location
UNDISC.DAT	Estimates of the remaining undiscovered non-associated conventional gas resource by geologic play	\RESOURCE\USDISC
UNCONV.DAT	Estimates of the remaining undiscovered non-associated unconventional gas resource by geologic play	\RESOURCE\USDISC
UNDOFF.DAT	Estimates of the remaining undiscovered offshore gas resource by geographical play	\RESOURCE\USDISC
REGION.DAT	File matching USGS plays containing supply regions and state/district codes	\RESOURCE\USUND
PLAYINFO.DAT	File of play averages for CO_2, H_2O, and N_2	\RESOURCE\USUND
THFAC.CON	File containing, for each play with undiscovered onshore conventional gas resource, the thickness for field size class reservoirs and the factor by which thickness increases across field size classes	\RESOURCE\USUND
THFAC.UNC	File containing, for each play with undiscovered onshore unconventional gas resource, the thickness for field size class reservoirs and the factor by which thickness increases across field size classes	\RESOURCE\USUND
THFAC.OFF	File containing, for each play with undiscovered offshore gas resource, the thickness for field size class reservoirs and the factor by which thickness increases across field size classes	\RESOURCE\USUND
FSCRESV.SPC	File containing technically recoverable reserve definitions by field size class and minimum field size class for which horizontal wells can be used for undeveloped reservoirs	\RESOURCE\USUND
UNDRES.CON	Play-specific dataset containing number of discovered reservoirs and undiscovered conventional resource (should be copied to \RESOURCE\USUND)	\RESOURCE\USDISC
UNDRES.UNC	Play-specific dataset containing number of discovered reservoirs and undiscovered unconventional resource (should be copied to \RESOURCE\USUND)	\RESOURCE\USDISC
UNDRES.OFF	Play-specific dataset containing number of discovered reservoirs and undiscovered offshore resource (should be copied to \RESOURCE\USUND)	\RESOURCE\USDISC
AVG.CON	Play-average dataset for undiscovered conventional reservoirs (should be copied to \RESOURCE\USUND)	\RESOURCE\USDISC
AVG.UNC	Play-average dataset for undiscovered unconventional reservoirs (should be copied to \RESOURCE\USUND)	\RESOURCE\USDISC
AVG.OFF	Play-average dataset for undiscovered offshore reservoirs (should be copied to \RESOURCE\USUND)	\RESOURCE\USDISC
PLY_DFN.TXT	File, containing estimates of the federal fraction of the resources of each undiscovered play	\RESOURCE\USUND

Program Files

Name	Description	Location
UNDSORT.SAS	SAS routine to sort final undiscovered conventional resource database	\RESOURCE\USUND
UNCSORT.SAS	SAS routine to sort final undiscovered unconventional resource database	\RESOURCE\USUND
OFFSORT.SAS	SAS routine to sort final undiscovered offshore resource database	\RESOURCE\USUND
UNDISC.EXE	FORTRAN routine which creates the final GSAM undiscovered conventional resource database	\RESOURCE\USUND
UNDUNC.EXE	FORTRAN routine which creates the final GSAM undiscovered unconventional resource database	\RESOURCE\USUND
UNDOFF.EXE	FORTRAN routine which creates the final GSAM undiscovered offshore resource database	\RESOURCE\USUND
FEDRES2.EXE	FORTRAN routine which splits GSAM undiscovered databases into Federal and Private databases	\RESOURCE\USUND
FEDRES.BAT	Batch file, which runs all undiscovered GSAM databases through the FEDRES2.EXE program to split them into federal, and private.	\RESOURCE\USUND

Intermediate Files

Name	Description	Location
UNDISC.INT	Final unsorted undiscovered reservoir database for conventional resource (generated by UNDISC.EXE)	\RESOURCE\USUND
UNDUNC.INT	Final unsorted undiscovered reservoir database for unconventional resource (generated by UNDUNC.EXE)	\RESOURCE\USUND
UNDOFF.INT	Final unsorted undiscovered reservoir database for offshore resource (UNDOFF.EXE)	\RESOURCE\USUND
UNDISC.GSM	GSAM Undiscovered Reservoir database for conventional resource (summary format, 1 line per reservoir, read by FEDRES2.EXE)	\RESOURCE\USUND
UNDCOL.GSM	GSAM Undiscovered Reservoir database for coalbed methane resource (summary format, 1 line per reservoir, read by FEDRES2.EXE)	\RESOURCE\USUND
UNDTGT.GSM	GSAM Undiscovered Reservoir database for tight resource (summary format, 1 line per reservoir, read by FEDRES2.EXE)	\RESOURCE\USUND
UNDOFF.GSM	GSAM Undiscovered Reservoir database for offshore resource (summary format, 1 line per reservoir, read by FEDRES2.EXE)	\RESOURCE\USUND

Output Files

UNDISCF.GSM	GSAM Undiscovered Reservoir database for federal conventional resource (summary format, 1 line per reservoir)	\RESOURCE\USUND
UNDISCP.GSM	GSAM Undiscovered Reservoir database for private conventional resource (summary format, 1 line per reservoir)	\RESOURCE\USUND
UNDCOLF.GSM	GSAM Undiscovered Reservoir database for federal coalbed methane resource (summary format, 1 line per reservoir)	\RESOURCE\USUND
UNDCOLP.GSM	GSAM Undiscovered Reservoir database for private coalbed methane resource (summary format, 1 line per reservoir)	\RESOURCE\USUND
UNDTGTF.GSM	GSAM Undiscovered Reservoir database for federal tight resource (summary format, 1 line per reservoir)	\RESOURCE\USUND
UNDTGTP.GSM	GSAM Undiscovered Reservoir database for private tight resource (summary format, 1 line per reservoir)	\RESOURCE\USUND
UNDOFFF.GSM	GSAM Undiscovered Reservoir database for offshore resource in the Gulf of Mexico-Central and Gulf of Mexico-West regions (summary format, 1 line per reservoir)	\RESOURCE\USUND
UNDGOME.GSM	GSAM Undiscovered Reservoir database for offshore resource in the Gulf of Mexico-East region (summary format, 1 line per reservoir)	\RESOURCE\USUND
UNDATL.GSM	GSAM Undiscovered Reservoir database for offshore resource in the Atlantic Offshore region (summary format, 1 line per reservoir)	\RESOURCE\USUND

C. Canada Resource Module

C.1. Discovered and Undiscovered Resource

Input Files

Name	Description	Location
POOLINFO.ASF	NRG pool identification, location, and general data	\RESOURCE\CANDISC RESOURCE\CANUND
POOLDISC.ASF	NRG pool discovery well data	\RESOURCE\CANDISC RESOURCE\CANUND
POOLCHAR.ASF	NRG pool rock and fluid characteristics data	\RESOURCE\CANDISC RESOURCE\CANUND
POOLOIL.ASF	NRG pool oil-in-place, production, and reserves data	\RESOURCE\CANDISC RESOURCE\CANUND
POOLGAS.ASF	NRG pool gas-in-place, production, and reserves data	\RESOURCE\CANDISC RESOURCE\CANUND
POOLNGL.ASF	NRG pool NGL, production, and reserves data	\RESOURCE\CANDISC RESOURCE\CANUND

POOLWELS.ASF	NRG pool wells data	\RESOURCE\CANDISC RESOURCE\CANUND
POOLSIZE.ASF	NRG pool total recovery data	\RESOURCE\CANDISC RESOURCE\CANUND
POOLMETH.ASF	NRG pool post-primary recovery methods data	\RESOURCE\CANDISC RESOURCE\CANUND

Program Files

Name	Description	Location
CANADA1.SAS	SAS routine to merge and format NRG databases	\RESOURCE\CANDISC RESOURCE\CANUND
CANADA2.SAS	SAS routine to merge and format NRG databases	\RESOURCE\CANDISC RESOURCE\CANUND
CANADA3.SAS	SAS routine to merge and format NRG databases	\RESOURCE\CANDISC RESOURCE\CANUND
CANADA4.SAS	SAS routine to merge and format NRG databases	\RESOURCE\CANDISC RESOURCE\CANUND
CANADA5.SAS	SAS routine to merge and format NRG databases	\RESOURCE\CANDISC RESOURCE\CANUND
CANJOIN.SAS	SAS routine which merges CANADA1.OUT through CANADA5.OUT to create CANADA.OUT	\RESOURCE\CANDISC RESOURCE\CANUND

Intermediate Files

Name	Description	Location
CANADA1.OUT	NRG raw pool characteristics and properties data	\RESOURCE\CANDISC RESOURCE\CANUND
CANADA2.OUT	NRG raw NGL production data	\RESOURCE\CANDISC RESOURCE\CANUND
CANADA3.OUT	NRG raw oil production data	\RESOURCE\CANDISC RESOURCE\CANUND
CANADA4.OUT	NRG raw pool size data	\RESOURCE\CANDISC RESOURCE\CANUND
CANADA5.OUT	NRG raw natural gas production data	\RESOURCE\CANDISC RESOURCE\CANUND
CANADA.OUT	Final database of NRG raw data	\RESOURCE\CANDISC RESOURCE\CANUND

C.2. Discovered Resource

Program Files

Name	Description	Location
CANAVG.SAS	SAS routine to calculate play averages for Canadian plays	\RESOURCE\CANDIS

	with discovered resource	
CANDISC.SAS	SAS routine to create GSAM formatted/defaulted database for Canada	\RESOURCE\CANDIS
CAN_ID.SAS	SAS routine to create a NRG ID to GSAM ID crosswalk	\RESOURCE\CANDIS

Intermediate Files

Name	Description	Location
CANAVG.DAT	Data file of play average properties for plays with discovered resource	\RESOURCE\CANDIS
CANXWALK.DAT	File to assign unique GSAM ID based on NRG ID	\RESOURCE\CANDIS

Output Files

Name	Description	Location
CANADA.GSM	Finalized GSAM known reservoir database (full format, 13 lines per reservoir)	\RESOURCE\CANDIS
CAN24.GSM	Reservoir database for Eastern Canada (GSAM Region 24) (full format, 13 lines per reservoir)	\RESOURCE\CANDIS
DAT_GSAM.CAN	Summary file of key reservoir data elements for each known Canadian reservoir	\RESOURCE\CANDISC
LOC_GSAM.CAN	Location file for each known Canadian reservoir	\RESOURCE\CANDISC
PRD_GSAM.CAN	Production summary file for each known Canadian reservoir	\RESOURCE\CANDISC

C.3. Undiscovered Resource

Input Files

Name	Description	Location
UNDISC.CAN	Estimates of the remaining undiscovered non-associated mature gas resource by play	\RESOURCE\CANUND
UNDISC.HYP	Estimates of the remaining undiscovered non-associated hypothetical gas resource by play	\RESOURCE\CANUND
REGION.CAN	File matching plays with mature undiscovered resource to provinces, supply regions, and state/district codes	\RESOURCE\CANUND
REGION.HYP	File matching plays with hypothetical undiscovered resource to provinces, supply regions, and state/district codes	\RESOURCE\CANUND
REGION.CCN	File matching plays with undiscovered coalbed methane resource to provinces, supply regions, and state/district codes	\RESOURCE\CANUND
REGION.TCN	File matching plays with undiscovered tight resource to provinces, supply regions, and state/district codes	\RESOURCE\CANUND
UNDRES.CCN	File containing the amount of undiscovered coalbed methane resource and the number of discovered reservoirs	\RESOURCE\CANUND

	by play	
UNDRES.TCN	File containing the amount of undiscovered tight resource and the number of discovered reservoirs by play	\RESOURCE\CANUND
AVG.CCN	File containing play averages for plays containing undiscovered coalbed methane resource	\RESOURCE\CANUND
AVG.TCN	File containing play averages for plays containing undiscovered coalbed methane resource	\RESOURCE\CANUND
THFAC.CAN	File containing, for each play with undiscovered conventional gas resource, the thickness for field size class reservoirs and the factor by which thickness increases across field size classes	\RESOURCE\CANUND
THFAC.HYP	File containing, for each hypothetical play with undiscovered gas resource, the thickness for field size class reservoirs and the factor by which thickness increases across field size classes	\RESOURCE\CANUND
FSCLAVG.SPC	File containing, the average coalbed methane resource by field size class	\RESOURCE\CANUND
THFAC.TCN	File containing, for each play with undiscovered tight gas resource, the thickness for field size class reservoirs and the factor by which thickness increases across field size classes	\RESOURCE\CANUND

Program Files

Name	Description	Location
CANUNDIS.SAS	SAS routine to create intermediate GSAM undiscovered resource databases for Canada (actual plays and hypothetical plays)	\RESOURCE\CANUND
CANUNAVG.SAS	SAS routine to calculate play averages for Canadian plays with undiscovered resource (actual plays and hypothetical plays)	\RESOURCE\CANUND
CANUNDSR.SAS	SAS routine to sort final undiscovered conventional resource database	\RESOURCE\CANUND
CANCCNSR.SAS	SAS routine to sort final undiscovered coalbed methane resource database	\RESOURCE\CANUND
CANTCNSR.SAS	SAS routine to sort final undiscovered tight resource database	\RESOURCE\CANUND
CANHYPSR.SAS	SAS routine to sort final hypothetical undiscovered resource database	\RESOURCE\CANUND
CANUNDIS.EXE	FORTRAN executable which creates the final GSAM undiscovered Canadian resource databases	\RESOURCE\CANUND
CANUNHYP.EXE	Routine to create the final GSAM undiscovered hypothetical resource database for Canada	\RESOURCE\CANUND
CANCOAL.EXE	Routine to create the final GSAM undiscovered coalbed methane resource database for Canada	\RESOURCE\CANUND
CANTIGHT.EXE	Routine to create the final GSAM undiscovered tight resource database for Canada	\RESOURCE\CANUND

Intermediate Files

Name	Description	Location
CANUNAVG.DAT	Data file of play average properties for plays with undiscovered resource, actual plays and hypothetical plays	\RESOURCE\CANUND
UNDRES.CAN	File containing the amount of undiscovered conventional resource and the number of discovered reservoirs by play	\RESOURCE\CANUND
UNDRES.HYP	File containing the amount of hypothetical resource	\RESOURCE\CANUND
AVG.CAN	File containing play averages for plays containing undiscovered conventional resource	\RESOURCE\CANUND
AVG.HYP	File containing play averages for plays containing undiscovered hypothetical resource	\RESOURCE\CANUND
UNDCAN.INT	Final unsorted undiscovered reservoir database for conventional reservoirs	\RESOURCE\CANUND
UNDCHYP.INT	Final unsorted undiscovered reservoir database for hypothetical reservoirs	\RESOURCE\CANUND
UNDTCN.INT	Final unsorted undiscovered reservoir database for tight reservoirs	\RESOURCE\CANUND
UNDCCN.INT	Final unsorted undiscovered reservoir database for coal reservoirs	\RESOURCE\CANUND

Output Files

Name	Description	Location
UNDCAN.GSM	GSAM undiscovered conventional reservoir database for mature Canadian resource (summary format, 1 line per reservoir)	\RESOURCE\CANUND
UNDCHYP.GSM	GSAM undiscovered reservoir database for hypothetical Canadian resource (summary format, 1 line per reservoir)	\RESOURCE\CANUND
UNDCCN.GSM	GSAM undiscovered reservoir database for Canadian coalbed methane resource (summary format, 1 line per reservoir)	\RESOURCE\CANUND
UNDTCN.GSM	GSAM undiscovered reservoir database for Canadian tight gas resource (summary format, 1 line per reservoir)	\RESOURCE\CANUND
SUM.CAN	Resource breakdown for each conventional play	\RESOURCE\CANUND
SUM.HYP	Resource breakdown for each hypothetical play	\RESOURCE\CANUND
SUM.TCN	Resource breakdown for each tight play	\RESOURCE\CANUND
SUM.CCN	Resource breakdown for each coalbed play	\RESOURCE\CANUND

D. Operational Procedures for Running the Resource Module

The entire Resource Module is run using Statistical Analysis System (SAS) and FORTRAN software. SAS was selected because of its ability to quickly complete required analytical procedures on large numbers of reservoirs with significant quantities of data. The program allows data to be sorted, merged, analyzed, and reported using program language similar to FORTRAN, but with more statistical capability.

Prior to running the Module, input files from NRG, USGS, Geological Survey of Canada, etc. must be prepared for analysis. These files must be in ASCII format. The NRG Associates data dictionary contains documentation on the data and formats in the reservoir data files. USGS and Geological Survey of Canada information includes the play identifier, NRG's corresponding cluster code, and the volume in billion cubic feet (Bcf) for the remaining resources.

The following two charts, Figures 2 and 3, show the process of creating the Resource Module’s output database files. As the database files have previously been created, and as any “branch” of the tree may be initiated at any point (with the assumption that all prerequisite files have been created), there is no single step-by-step operational procedure. Instead, following the chart from where a new data input file has been created or altered to the desired new output file is the manner in which the Module truly operates.

Figure 2
U.S. Resource Module

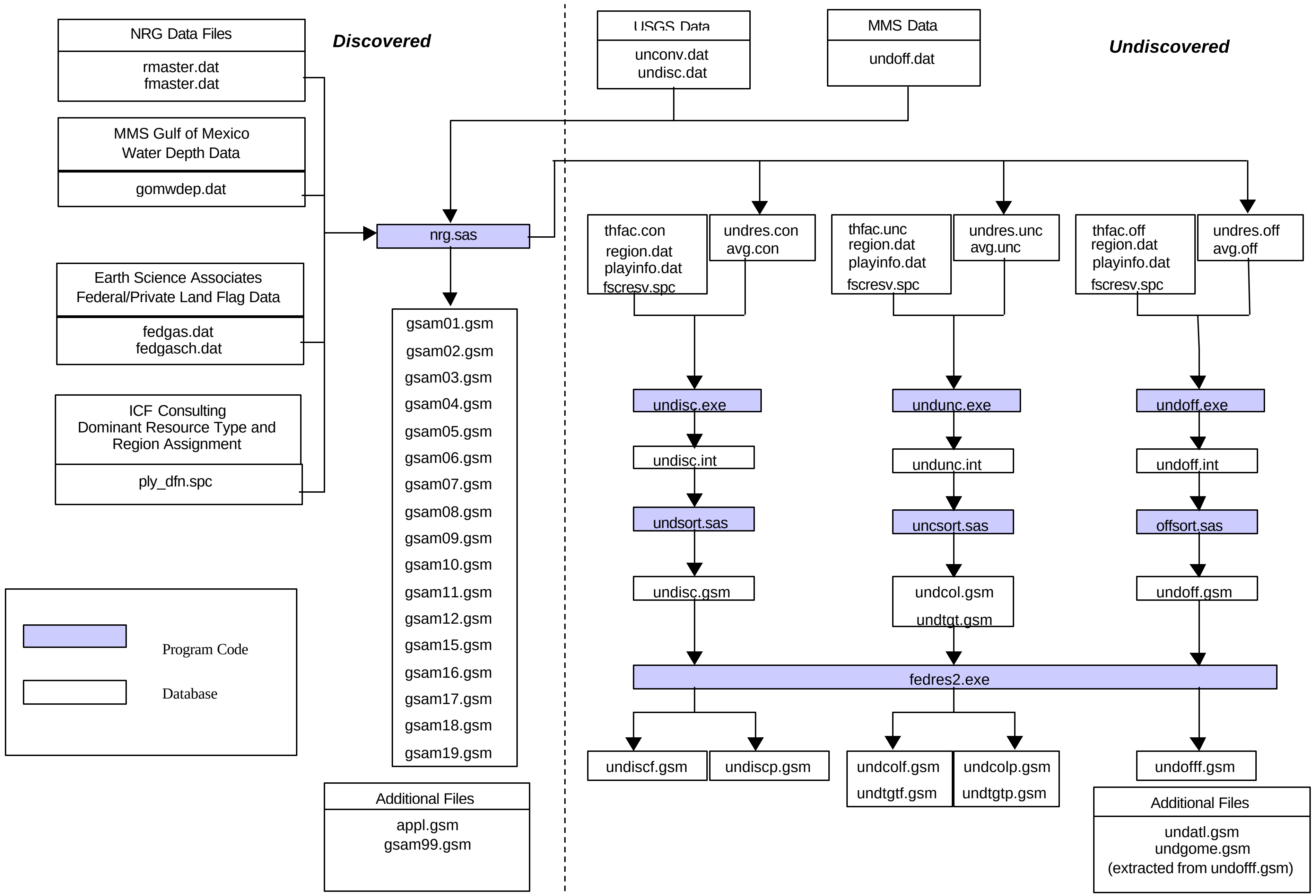
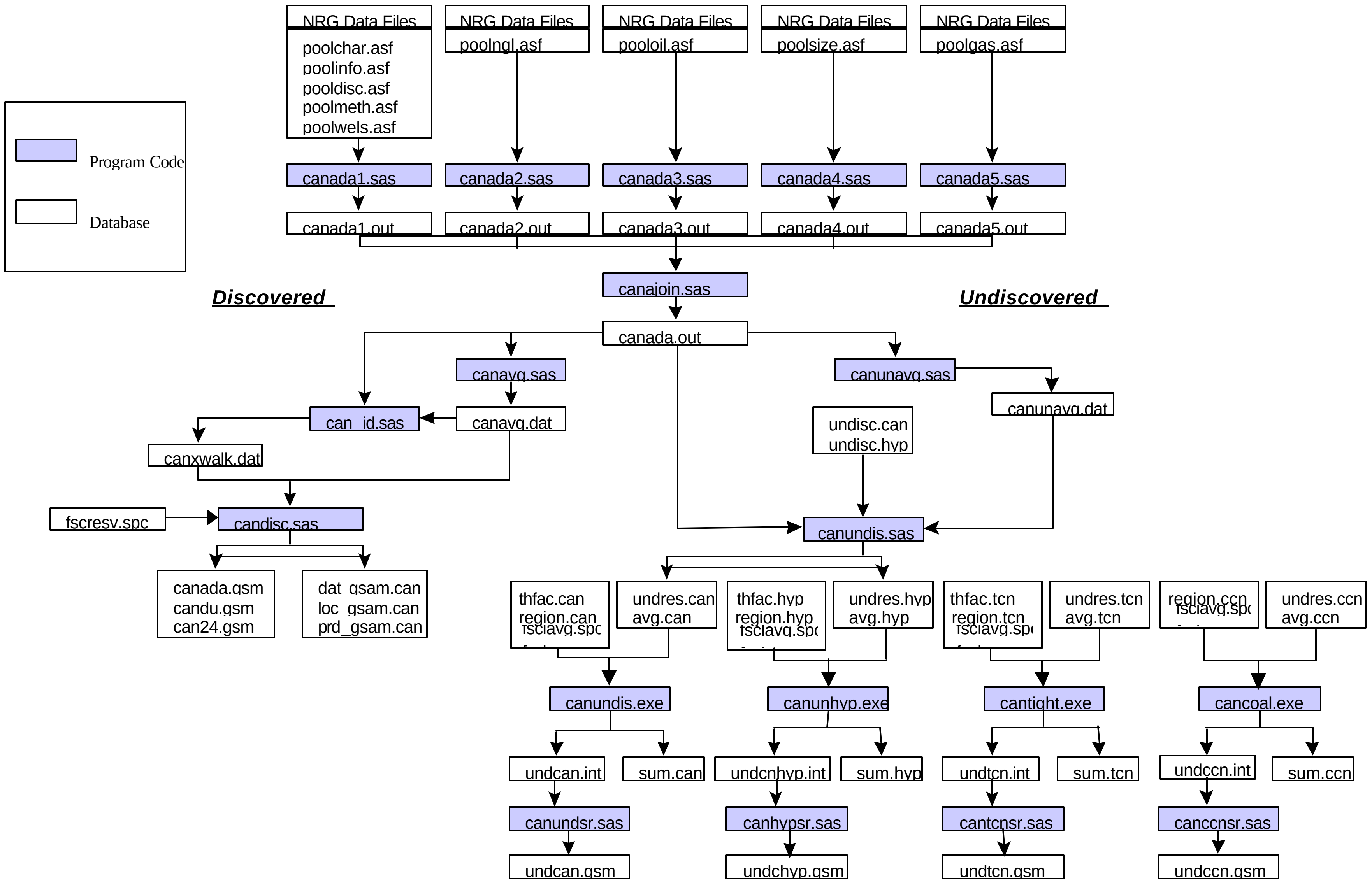


Figure 3
Canadian Resource Module



E. Description of Module Components

The GSAM Resource Module can be conceptually divided into three segments. The first segment reads and merges files from the NRG data sets into a consistent form. The second creates the known (discovered) database(s) based on the reservoirs in the NRG data. The final segment creates the undiscovered database based on USGS, ICF Consulting, MMS, Geological Survey of Canada, Alberta Geological Survey, NPC resource estimates, and NRG data.

Data inputs for the model currently consist of files from the NRG Associates Database of Significant Oil and Gas Fields of the United States (production specified up to 1997) and NRG Associates Significant Oil and Gas Pools of Canada (production specified up to 1994). Specifically, data in the RMASTER.DAT and FMASTER.DAT files (which contain data compiled straight from NRG) are used in the Module to create the U.S. GSAM Reservoir Databases. Data form NRG Associates Significant Oil and Gas Pools of Canada in the POOLINFO.ASF, POOLCHAR.ASF, POOLDISC.ASF, POOLOIL.ASF, POOLGAS.ASF, POOLLNG.ASF, POOLSIZE.ASF, POOLMETH.ASF, and POOLWELS.ASF files is used to create the Canadian GSAM Reservoir Databases. These databases are read by the SAS routines and analyzed to assure internal consistency.

The U.S. database relies on USGS play codes (4-digit alpha-numeric play code) as the basis for play analysis. The Canadian database relies on NRG Associates cluster codes as the basis for play analyses. Data from the Geological Survey of Canada covering the remaining undiscovered resource in various basins and plays were allocated to NRG clusters by matching the Geological Survey basin and play names to NRG play names.

Play average calculations for key reservoir properties are the first order procedure for calculating default parameters for both U.S. and Canadian reservoirs. The process considers all non-zero values in the database for porosity, permeability, depth, reservoir temperature, initial pressure, and gas specific gravity. Temperature and pressure values are further converted to gradients based on depth. These values serve two functions in GSAM. They serve as defaults for known reservoirs in each play that do not have values in the NRG databases, and they are used in describing the undiscovered reservoirs in the play.

Additional default procedures are also used in assessing missing data elements. Internal consistency checks have been developed to validate and appropriately adjust key reservoir parameters. Volumetric parameters including porosity, initial fluid saturations, area, thickness, and formation volume factors are analyzed and compared to resource in-place and cumulative recovery values in the database. Standard relationships between porosity and permeability are also used to estimate a missing value when only one is provided in the NRG data. Finally, regional and national defaults are provided for some properties, including Langmuir pressure, desorption, and gas content for coals and shales. These defaults provide a reasonable representation of the reservoir properties when no other data is available.

Additionally, the reservoirs in the offshore fields of the Gulf of Mexico regions are stacked using the following procedure:

(a) Aggregate GOM offshore reservoirs into twelve geographic plays (and not geological play) based on water depth:

Table 2
Water Depth Aggregation

Water Depth			**GSAM PLAY CODE**		
(meter)	**(feet)**	**Average (feet)**	**GOM-C**	**GOM-W**	**GOM-E**
0-60	0-196.8	98.4	9901	9905	9921
60-200	196.8-656.2	426.5	9902	9906	9909
200-900	656.2-2952.7	1804.5	9903	9907	9923
>900	>2952.7	5000	9904	9908	9910

Note: GOM-C= Gulf of Mexico Central
GOM-W= Gulf of Mexico West
GOM-E= Gulf of Mexico East

(b) Aggregate reservoirs located within the same NRG cluster/field into a single reservoir according to the following guidelines:

- Reservoir acreage is set to the largest reservoir acreage in the field.
- Well depth is set to the deepest reservoir well depth in the field.
- Water depth is set to the deepest reservoir water depth in the field.
- Field level data is utilized for Original Gas In Place (OGIP), gas production, oil production, NGL production, total number of wells, number of active wells, and number of inactive wells.
- Volumetric average is used permeability, porosity, initial gas and water saturations, initial pressure, gas specific gravity, bottom hole temperature, CO_2, N_2, and H_2S concentration, and gas Z-factor.

- Other reservoir properties are set equal to the properties of the largest acreage reservoir in the field.

A unique 12-digit GSAM identification number is assigned to each reservoir. This twelve-digit number consists of six individual identifying factors:

Region	Status	Dominant Resource Type of Play	Play	Federal/Private Land Flag	Resv. Number	GSAMID
07	3	1	4401	P	001	==> 07314401P001

This describes a reservoir in the Permian GSAM region (region: 07), that is currently producing (status: 03), is a conventional formation (resource: 1), is in USGS play number 4401, and is on private land (flag: P). The reservoir number provides a unique identifier for each reservoir in a play. In undiscovered reservoirs, the status is listed as 1, and the reservoir number identifies the size class of the reservoir. Appendix A contains dictionaries of regions, status, and resource type. Note that the GSAMID-s of all Canadian reservoirs are manually updated to 12-digit (by inserting "p" at the end of the play ID) for consistency purposes. The Canadian Resource Module creates 11-digit GSAMID-s.

Based on the input data, play average defaults, internal consistency checks, and regional/national defaults, known reservoirs are fully characterized in GSAM. Table A-1 of Appendix A provides the full structure of the discovered reservoir database for the U.S. (GSAM##.GSM files), including the type of data element (text, integer, real, etc.). It consists of 190 individual elements. Table A-2 of Appendix A provides the format of the known Canadian reservoir database.

The undiscovered resource characterization in GSAM relies on estimates from the USGS, MMS, previous ICF work, Geological Survey of Canada, Alberta Geological Survey, and NPC to provide volumes of the total undiscovered non-associated gas resource. Estimates are established for each play defined in GSAM. These values are transformed into reservoirs of various sizes based on the play averages for key reservoir properties in each play. The area and thickness of reservoirs in each size class of undiscovered reservoirs is calculated based on the average gas-in-place for the class and the properties and area-thickness relationships developed by analyzing the discovered resource. The resulting files, UNDISC.GSM, UNDCOL.GSM, UNDTGT.GSM, UNDOFF.GSM, UNDCAN.GSM, UNDCHYP.GSM, UNDCCN.GSM, and UNDTCN.GSM are formatted as a single line of data for each reservoir.

A FORTRAN program (FEDRES2.EXE) is used to split GSAM undiscovered database into Federal and Private Land databases. The splitting process is performed to calculate number of undiscovered accumulations (NRR) in each field size class (FSC) based on average recoverable reserve fraction of Federal land in the corresponding play. In GSAM, a play in the undiscovered resource is defined as a group of 13 field size classes (FSC 5 to FSC 17). The following steps are carried out for every play in the GSAM undiscovered databases to split the NRR of each FSC:

(c) Read reservoir properties and NRR of 13 FSC records (of one play) from undiscovered GSAM database (*.GSM). Based on play level recovery factor obtained from play average property file (AVG.*), calculate average and total reserves of each FSC in the play. Table 2 shows USGS play "2212" in San Juan region (GSAM region 09) from GSAM database file UNDISC.GSM. Notice that each FSC record is indicated by 11-digit GSAMID.

(d) Read play level Federal fraction obtained from play definition file (PLY_DFN.TXT). For the example in Step 1, the corresponding undiscovered Federal fraction from PLY_DFN.TXT is 0.5.

(e) Apply the Federal fraction to the total of NRR in each FSC to get the first estimate of NRR for Federal land and Private land. First, calculate the Federal NRR by taking the integer part of the product of Federal fraction and total NRR. The Private NRR is then set to the remaining NRR in the FSC. Using the calculated FSC average reserves (Table 2), calculate FSC and total Federal and Private reserves. Table 3 shows NRRs and reserves of Federal and Private lands. The bottom row of Table II-5 is the calculated Federal and Private reserve fractions. Notice that the calculated Federal fraction (0.32) is different with the data obtained from PLY-DFN.TXT (0.5). This results from rounding of NRR into an integer.

(f) Adjust the Federal and Private NRRs of each FSC by subtracting or adding one accumulation from the NRRs to get the best possible estimate of Federal and Private NRRs. The adjustment is done by calculating Federal reserve fractions of 8192 combinations (i.e. 2^{13}) for every play, and select one combination that gives the closest Federal fraction to the data read from PLY_DFN.TXT (which in this case is 0.5). Table 4 shows the final NRR splitting calculation that gives smallest deviation between calculated and expected Federal fractions (within 2% error).

Table 3
FSC Data and Calculated Reserves of an Undiscovered Play

GSAM ID	FSC	NRR	Avg. Reserve (BCF)	Total Reserve (BCF)
09112212005	5	9	4.5	40.5
09112212006	6	5	9.0	45.0
09112212007	7	3	18.0	54.0
09112212008	8	2	36.0	72.0
09112212009	9	1	72.0	72.0
09112212010	10	0	144.0	0.0
09112212011	11	0	288.0	0.0
09112212012	12	0	576.0	0.0
09112212013	13	0	1152.0	0.0
09112212014	14	0	2304.0	0.0
09112212015	15	0	4608.0	0.0
09112212016	16	0	9216.0	0.0
09112212017	17	0	18432.0	0.0
Total for play "2212"		20		283.5

Table 4
First Estimate of Federal NRR and Private NRR (Federal Fraction=0.5)

FSC	NRR	Federal NRR	Private NRR	Federal Reserve (BCF)	Private Reserve (BCF)
5	9	4	5	18.0	22.5
6	5	2	3	18.0	27.0
7	3	1	2	18.0	36.0
8	2	1	1	36.0	36.0
9	1	0	1	0.0	72.0
10	0	0	0	0.0	0.0
11	0	0	0	0.0	0.0
12	0	0	0	0.0	0.0
13	0	0	0	0.0	0.0
14	0	0	0	0.0	0.0
15	0	0	0	0.0	0.0
16	0	0	0	0.0	0.0
17	0	0	0	0.0	0.0
Total	20	8	12	90.0	193.5

Calculated Reserve Fraction	0.32	0.68
Correct Reserve Fraction	0.50	0.50

Table 5
Final Estimate of Federal NRR and Private NRR (Federal Fraction=0.5)

FSC	Federal			Private		
	GSAM ID	NRR	Reserve (BCF)	GSAM ID	NRR	Reserve (BCF)
5	09112212F005	5	22.5	09112212P005	4	18.0
6	09112212F006	3	27.0	09112212P006	2	18.0
7	09112212F007	1	18.0	09112212P007	2	36.0
8	09112212F008	2	72.0	09112212P008	0	0.0
9	09112212F009	0	0.0	09112212P009	1	72.0
10	09112212F010	0	0.0	09112212P010	0	0.0
11	09112212F011	0	0.0	09112212P011	0	0.0
12	09112212F012	0	0.0	09112212P012	0	0.0
13	09112212F013	0	0.0	09112212P013	0	0.0
14	09112212F014	0	0.0	09112212P014	0	0.0
15	09112212F015	0	0.0	09112212P015	0	0.0
16	09112212F016	0	0.0	09112212P016	0	0.0
17	09112212F017	0	0.0	09112212P017	0	0.0
Total		11	139.5		9	144.0

Calculated Reserve Fraction	0.49	0.51
Correct Reserve Fraction	0.50	0.50

(g) Create two GSAM database files, one for Federal land and one for Private land, and store the final NRR values with the same reservoir properties as in the original GSAM database. In these two files a letter "F" for Federal portion or "P" for Private portion is inserted after the 8th character of the original GSAMID (see Table 4). For UNDISC.GSM, the Federal land database file will be named UNDISCF.GSM and the Private land database file will be named UNDISCP.GSM. In the case when there is no Private land is found in the original database (e.g. UNDOFF.GSM for undiscovered offshore GSAM database), zero size Private land database will be created. This file should not be used in any GSAM run.

The resulting files, UNDISCF.GSM, UNDISCP, UNDCOLF.GSM, UNDCOLP.GSM, UNDTGTF.GSM, UNDTGTP.GSM, UNDOFFF.GSM, UNDCAN.GSM, UNDCHYP.GSM, UNDCCN.GSM, and UNDTCN.GSM are formatted as a single line of data for each reservoir. Appendix A, Table A-7 shows the format for these files. The files UNDATL.GSM and UNDGOME.GSM, which contain offshore reservoirs in the Atlantic Offshore and GOM-E regions respectively, are extracted from the file UNDOFFF.GSM, which contains offshore reservoirs in the GOM-C and GOM-W regions only.

V. Reservoir Performance Module (2)

A. Summary Description of the Reservoir Performance Module

The Reservoir Performance Module develops reservoir production response estimates and summary project economics based on the reservoir data output from the Resource Module, and input on technology specifications, resource-specific requirements, regional costs and operating parameters, state and federal tax requirements, and other assumptions. The production response estimates and project economics are subsequently used by other modules of GSAM.

The type curve modules and costing routines of the Reservoir Performance Module generate production and cost responses, which are passed to the Exploration and Production Module of GSAM. The Reservoir Performance Module generates output under assumptions for both current and advanced drilling, completion, and costing technology scenarios. In addition, the .SUM (for current technology) and .ASM files (for advanced technology) files created from the Reservoir Performance Module can be used to create price supply curves (MASP versus reserves).

B. Required Files

All reservoir data used by the Reservoir Performance Module comes from the formatted and checked files output by the Resource Module. These files are divided by region, and are listed below:

B.1 Reservoir Database Files

B.1.1 United States

Discovered Reservoir Database

Name	Description	Location
GSAM##.GSM	Known reservoirs in the GSAM supply regions ("##" is a two-digit code, ranging from 1 to 12 and from 15 to 19; see table A-8 from Appendix A for supply region names and codes	\RESVPERF
GSAM99.GSM	GSAM pseudo discovered reservoirs database, developed to account for NRG shortfall in production and reserves	\RESVPERF
APPL.GSM	Appalachian reservoirs (GSAM region 01)	\RESVPERF

Undiscovered Reservoir Database

Name	Description	Location
UNDISCF.GSM	GSAM Undiscovered Reservoir database for federal conventional resource (summary format, 1 line per reservoir)	\RESVPERF
UNDISCP.GSM	GSAM Undiscovered Reservoir database for private conventional resource (summary format, 1 line per reservoir)	\RESVPERF
UNDCOLF.GSM	GSAM Undiscovered Reservoir database for federal coalbed methane resource (summary format, 1 line per reservoir)	\RESVPERF
UNDCOLP.GSM	GSAM Undiscovered Reservoir database for private coalbed methane resource (summary format, 1 line per reservoir)	\RESVPERF
UNDTGTF.GSM	GSAM Undiscovered Reservoir database for federal tight resource (summary format, 1 line per reservoir)	\RESVPERF
UNDTGTP.GSM	GSAM Undiscovered Reservoir database for private tight resource (summary format, 1 line per reservoir)	\RESVPERF
UNDOFFF.GSM	GSAM Undiscovered Reservoir database for offshore resource in GOM-C and GOM-W regions (summary format, 1 line per reservoir)	\RESVPERF
UNDATL.GSM	GSAM Undiscovered Reservoir database for offshore resource in Atlantic Offshore region (summary format, 1 line per reservoir)	\RESVPERF
UNDGOME.GSM	GSAM Undiscovered Reservoir database for offshore resource in GOM-E region (summary format, 1 line per reservoir)	\RESVPERF

B.1.2 Canada

Discovered Reservoir Database

Name	Description	Location
CANADA.GSM	Known Canadian Reservoirs Alberta, (region: 22) British Columbia (region: 23)	\RESVPERF

Undiscovered Reservoir Database

Name	Description	Location
UNDCHYP.GSM	Undiscovered Canadian Reservoirs (hypothetical plays)	\RESVPERF
UNDCAN.GSM	Undiscovered Canadian Reservoirs (conventional plays)	\RESVPERF
UNDCCN.GSM	Undiscovered Canadian Reservoirs (coalbed methane)	\RESVPERF
UNDTCN.GSM	Undiscovered Canadian Reservoirs (tight)	\RESVPERF

Undeveloped Reservoir Database

Name	Description	Location
CANDU.GSM	Discovered undeveloped Canadian reservoirs	\RESVPERF

B.2 Data Input Files

The other data and assumptions required to run the Module, including technology specifications, regional production costs, and state and federal tax requirements are contained in data files which are also read into the Reservoir Performance Module and listed below:

Name	Description	Location
COST.VUS	Regional and resource-specific costs and investment data for U.S. vertical wells; must be copied to COST.DAT if used for the analysis	\RESVPERF\DATA
COST.VCN	Regional and resource-specific costs and investment data for Canadian vertical wells; must be copied to COST.DAT if used for the analysis	\RESVPERF\DATA
COST.HUS	Regional and resource-specific costs and investment data for U.S. horizontal wells; must be copied to COST.DAT if used for the analysis	\RESVPERF\DATA
COST.HCN	Regional and resource-specific costs and investment data for Canadian horizontal wells; must be copied to COST.DAT if used for the analysis	\RESVPERF\DATA
TECH.VER	Technology parameters for vertical wells in both the U.S. and Canada; must be copied to TECH.DAT if used for the analysis	\RESVPERF\DATA
TECH.HOR	Technology parameters for horizontal wells in both the U.S. and Canada; must be copied to TECH.DAT if used for the analysis	\RESVPERF\DATA
TECH.APL	Technology parameters for Appalachian discovered reservoirs (APPL.GSM); must be copied to TECH.DAT if used for the analysis	\RESVPERF\DATA
TAXES.DAT	Severance, income, and ad valorem taxes by State/District	\RESVPERF\DATA
TAX_NAT.DAT	Federal tax specifications and other tax structures and costs for U.S and Canada.	\RESVPERF\DATA
AFE.DAT	Percentages of investments into normal AFE categories	\RESVPERF\DATA
PLY_DFN.SPC	Dominant Resource Type, Region Play Assignments, Exploration depths, etc. (used by Exploration and Production Module and Resource Module)	\EXPLPROD
GEOLOGY.DAT	Specifies reservoir properties by pay-grade distribution	\RESVPERF\DATA
TEMPLATE.DAT	File containing description of type curve input parameters, must be specified in REGIONS.DAT to create .TCI files	\RESVPERF\DATA

It should be noted that the user may create or modify any of these input files. The Module's code reads the numbers in the Reservoir Performance data files as free-format so the position, decimal places, etc. of the data does not have to be specified at a particular location in the file. Note also that data files such as COST.DAT have headers throughout the data, and whether these files are modified or recreated, these headers must remain in the same format. Finally, when creating new data files, be sure that the DOS file name is the same as the original (with a different extension), and the file to be used is copied so that its extension is .DAT (or .HOR for the horizontal well case). Unlike the data files, run specification files, described below, do need to have exact formatting.

B.3 Run Specification Files

The following files contain instructions on the type and configuration of the Reservoir Performance run being conducted. They are read by the FORTRAN executable program to set up the formats, identify inputs, and set up key parameters. The files include:

Name	Description	Location
REGIONS.01-REGIONS18	Specifies the discovered U.S. reservoir data sets (GSAM01.GSM onwards) for the analysis; must be copied to REGIONS.DAT if used for the analysis	\RESVPERF
REGIONS.CAN	Specifies the discovered Canadian reservoir data set (CANADA.GSM) for the analysis; must be copied to REGIONS.DAT if used for the analysis	\RESVPERF
REGIONS.APL	Specifies the Appalachian reservoir data set for the analysis; must be copied to REGIONS.DAT if used for the analysis	\RESVPERF
REGIONS.CDU	Specifies the Canadian discovered, undeveloped reservoir data set for the analysis; must be copied to REGIONS.DAT if used for the analysis	\RESVPERF
REGIONS.UND	Specifies the U.S. undiscovered reservoir data sets for the analysis; must be copied to REGIONS.DAT if used for the analysis	\RESVPERF
REGIONS.UNC	Specifies the Canadian undiscovered reservoir data sets for the analysis; must be copied to REGIONS.DAT if used for the analysis	\RESVPERF
RUNSET.DIS	Run specifications (start year, analysis type, environmental parameters, etc.) for the U.S. discovered reservoir data sets; must be copied to RUNSET.DAT if used for the analysis	\RESVPERF
RUNSET.CAN	Run specifications (start year, analysis type, environmental parameters, etc.) for the discovered Canadian reservoir data set; must be copied to RUNSET.DAT if used for the analysis	\RESVPERF
RUNSET.APL	Run specifications (start year, analysis type, environmental parameters, etc.) for the Appalachian reservoir data set; must be copied to RUNSET.DAT if used for the analysis	\RESVPERF
RUNSET.CDU	Run specifications (start year, analysis type, environmental parameters, etc.) for the Canadian discovered, undeveloped reservoir data set; must be copied to RUNSET.DAT if used for	\RESVPERF

	the analysis	
RUNSET.UND	Run specifications (start year, analysis type, environmental parameters, etc.) for the U.S. undiscovered reservoir data sets; must be copied to RUNSET.DAT if used for the analysis	\RESVPERF
RUNSET.UNC	Run specifications (start year, analysis type, environmental parameters, etc.) for the Canadian undiscovered reservoir data sets; must be copied to RUNSET.DAT if used for the analysis	\RESVPERF

B.4 Program Files

Name	Description	Location
RESVPERF.EXE	FORTRAN executable which will run the Reservoir Performance Module	\RESVPERF
PERFRESV.BAT	Batch file which will automate the module for a specified data input file and given run specification file	\RESVPERF
RUNALLRP.BAT	Batch file which will automate the module for all regions and all resource types	\RESVPERF

Appendix B contains printouts of the input data and run specification files currently being used in the GSAM Reservoir Performance Module. Again, these files can be modified to conduct individual analyses by changing the resource/reservoir performance parameters, and/or significant changes in development or operating costs.

C. Output Files

This Module will create the following main output files. To ensure that these files are not overwritten in a subsequent run of the Module, be sure to copy these files to another file name or directory:

Name	Description	Location
.DEC	Files for E&P decisions including summary economics (one corresponding to each .GSM input file)	\RESVPERF
.PRD	Files summarizing annual production and operating costs (one corresponding to each .GSM input file)	\RESVPERF
.ENV	Files for use in the Environmental Module (one corresponding to each .GSM input file)	\RESVPERF

In addition, summary files are created to assist in quality control/quality assurance. These include:

Name	Description	Location
.CUR	Summary of current technology case results, including the Minimum Acceptable Supply Price (MASP) (for pay grade 2), well depth, total production, and wells (one .CUR file corresponding to each .GSM input file)	\RESVPERF
.ADV	Summary of advanced technology case results, including MASP (for pay grade 2), well depth, production, and wells (one .ADV file corresponding to each .GSM input file)	\RESVPERF
.SUM	Summary of MASP and reserves by pay grade for current technology case results (one .SUM corresponding to each .GSM input file. NOTE: Can be used to create price-supply curve for current technology)	\RESVPERF
.ASM	Summary of MASP and reserves by pay grade for advanced technology case results (one .ASM file corresponding to each .GSM input file. NOTE: Can be used to create price-supply curve for advanced technology)	\RESVPERF

Additional reservoir-level summary files are created if requested in REGIONS.DAT. These include:

.PRO	Optional output file containing detailed cash flow pro-forma on a reservoir level; it is created only if its flag in REGIONS.DAT is set to “YES”	\RESVPERF
.TCO	Optional output file containing type curve output on a reservoir level; it is created only if its flag in REGIONS.DAT is set to “YES”	\RESVPERF
.NPV	Optional output file containing net present value summary on a reservoir level; it is created only if its flag in REGIONS.DAT is set to “YES”	\RESVPERF
.PRR	Optional output file containing reduced form cash flow pro-forma on a reservoir level; it is created only if its flag in REGIONS.DAT is set to “YES”	\RESVPERF
.TCI	Optional output file containing type curve input on a reservoir level; it is created only if its flag in REGIONS.DAT is set to “YES”	\RESVPERF

C.1 Associated Gas and Mexico Production Files

Additional .PRD and .DEC files are created for associated gas (ADGAS) and for Mexican gas production to allow for testing and full system analysis. Two Mexico .PRD files are used to characterize Mexico importing gas from the United States and Mexico exporting gas to the United States. These files contain aggregated regional production data, not based on reservoir by reservoir analysis, and are located in the \GSAM\EXPLPROD\TMP directory. The ADGAS file is created from DOE's Crude Oil Policy Model (COPM).

D. Operational Procedures for Running the Reservoir Performance Module

In the following section we describe two options for running the Reservoir Performance Module: (1) manual operation and (2) automation through batch files. The batch files combine many of the manual steps, and will produce exactly the same results as the manual operation, if both are run correctly. Manual operation should be used in testing (for example, to experiment with some changed data parameter in cost or technology) or to identify exactly where problems occur if the model is not running properly. Batch files should be used in most other cases, as performing a complete run using the manual steps would be a repetitive, time-consuming, and error-prone activity. Instructions are provided so that the Module can be run as a stand-alone component or as a piece of a fully integrated GSAM run encompassing all modules.

Before beginning a Reservoir Performance run, make certain that all input files, including the .GSM files from the Resource Module, are in the correct location (use Section B above as a reference). A typical Reservoir Performance run contains two alternative technology cases, current and advanced. If this is to be changed, it is done so in COST.DAT and TECH.DAT. Although the Reservoir Performance Module is usually run with all regions and resource types, it can be run with any number of regions, hence, a preliminary decision must be made as to which region or regions are desired in the analysis. To customize the run at the regional level, be sure that only the desired .GSM files are listed in the appropriate REGIONS.XXX file, which will be copied to REGIONS.DAT. This is done where the option to run the type curve model is set (for more information on type curves, see Section E below).

REGIONS.DAT contains options to print out a variety of files. These files are for debugging and should ordinarily not be printed ("NO"). These files show data from individual reservoirs, so if they are activated, the files will grow extremely large if a full RP run is being performed. These files are described in the Appendix. There is also an indicator of the number of years that the RP Module will run with, which should be 40, and should not be altered. The time frame of the analysis is set in the Exploration and Production and Demand and Integrating Modules, not in the RP Module. Having a 40-year time horizon for reservoir performance allows the model to estimate future production, and the future revenue stream, of any reservoir that will be analyzed in the feasible time period, even if some of its future production is outside the, for example, 20-year time frame set in the E&P and D&I Modules.

There are tax, cost, and technology (for Appalachia) files associated with some of the GSAM regions, so a decision must be made as to which tax, cost, and technology scenario is appropriate for the run. Finally, the run options in RUNSET.XXX must be set as desired. These options are explained in the Appendix.

To run the Reservoir Performance Module with only one region with one resource type, for example, conventional undiscovered reservoirs in Canada, REGIONS.UNC must be modified so that UNDCAN is the only region listed, and REGIONS.UNC must be copied to REGIONS.DAT. If RUNSET.UNC does not contain the desired specifications for this run, it must also be modified before being copied to RUNSET.DAT. To carry out this run with Canadian vertical well costs and Canadian taxes, copy COST.VCN to COST.DAT and TECH.VER to TECH.DAT. The Module is now prepared to be run with this single region under both current and advanced technology. The user may also edit the files (such as COST.DAT, TAX_NAT.DAT, etc.) and change the entries for creating special scenarios.

D.1 Manual Operation

For any scenario, the following gives step by step directions to manually running the Reservoir Performance Module:

a) Select a REGIONS file and copy the appropriate (or desired) COST.XXX to COST.DAT and the appropriate TECH.XXX to TECH.DAT.

   NOTE: Any combination of region, resource characterization, cost, and technology parameters may be selected for analysis. However, .GSM files have two different formats, 1-line format or 13-line format, and these two formats cannot be combined in the REGIONS file. Several .GSM files of the same format may be combined in any region file, specifying that format in the RUNSET file.

b) Copy the desired REGIONS.XXX file to REGIONS.DAT and copy the corresponding RUNSET.XXX to RUNSET.DAT, modifying these run specification files as desired

   NOTE: At this point, a decision may be made on whether to utilize the type curve model, set in REGIONS.DAT. Running without the type curve will significantly reduce the Module's run time (see section E-2 below for the function of the type curve model). The *.BIN files must be generated from the type curve process before running without the type curves). A run without type curves may be performed when only economic parameters have changed.

c) Run RESVPERF.EXE

d) Copy the .SUM, .ASM, .DEC, .PRD, and .ENV output files to GSAM\EXPLPROD\TMP directory. For undiscovered .GSM files and Discovered Undeveloped .GSM files (such as CANDU.GSM), copy the .SUM, .ASM, .DEC, .PRD, and .ENV files in GSAM\EXPLPROD\VERTICAL directory for vertical wells run. For horizontal wells run these files are copied into GSAM\EXPLPROD\HORZ directory

e) Repeat steps a - d until output files for all desired regions and resource types have been created

NOTE: GSAM can model both horizontal and vertical development technologies with the RP Module. Use the appropriate batch files (or use the horizontal COST and TECH files manually) to create horizontal performance files in addition to the vertical well performance file. Once the RP processing is complete, a routine in the E&P Module will select the best (horizontal vs. vertical) drilling alternative. See "Operational Procedures for running the Exploration and Production Module" in the next chapter.

NOTE: An integrated run requires that all regions be included, so for an integrated run, process every region through the Reservoir Performance Module (see Chapter VII, "Demand and Integrating Modules", for more details). Also note that a run that will be submitted to the Production Accounting Module requires both vertical and horizontal wells processing.

D.2 Batch File Operation

Six batch files are available to run the Reservoir Performance Module under either vertical only or horizontal and vertical drilling well parameters. These files are:

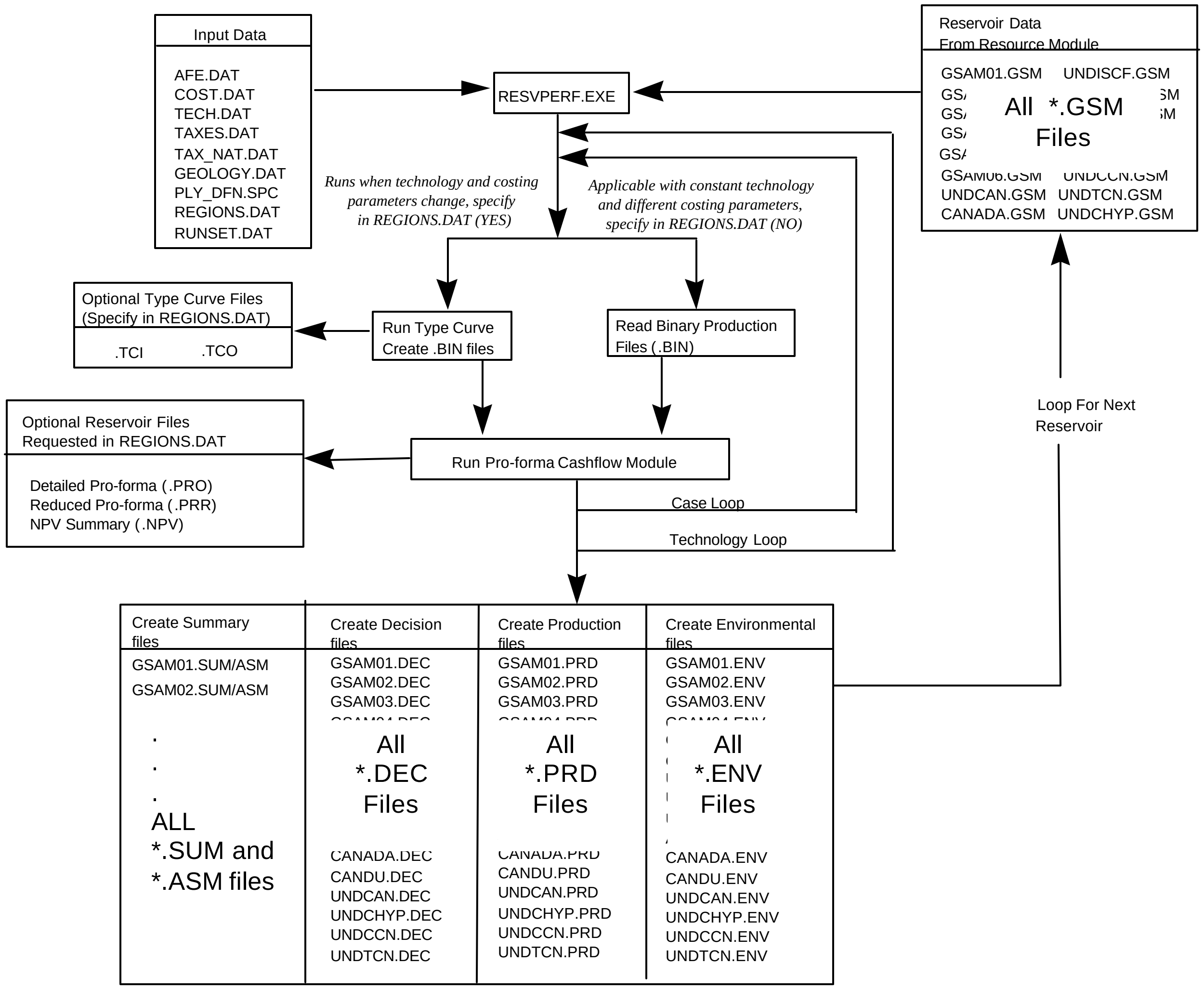
1) DISNRG.BAT
2) APPL.BAT
3) UNDUS.BAT
4) CANADA.BAT
5) CANDU.BAT
6) UNDCAN.BAT

DISNRG.BAT will run the U.S. discovered (i.e. NRG specified) reservoirs through the RP Module with vertical wells only, and will place the output in the GSAM\EXPLPROD\TMP directory. UNDUS.BAT and UNDCAN.BAT will run the U.S. and Canadian Undiscovered reservoirs with both vertical and horizontal technology, placing the files into position to be ready for the selection routine in the E&P Module (they are copied into the GSAM\EXPLPROD\TMP\VERTICAL and GSAM\EXPLPROD\TMP\HORZ directories). APPL.BAT file runs Appalachian discovered reservoirs through the RP module and copies the file in GSAM\EXPLPROD\TMP directory. CANADA.BAT file runs discovered producing reservoirs through the RP module and copies the file in GSAM\EXPLPROD\TMP directory. CANDU.BAT file runs discovered undeveloped Canadian reservoirs and copies files in GSAM\EXPLPROD\TMP\VERTICAL and GSAM\EXPLPROD\TMP\HORZ directories. RUNALLRP.BAT will run the Reservoir Performance Module with all regions and all resource characterizations. This batch file is the ideal procedure if a standard fully integrated GSAM run is desired. It runs all the GSAM regions through the Reservoir Performance Module with both vertical and horizontal well technology, copies the appropriate files into the Exploration and Production Module subdirectories, and creates binary reservoir bank files and environmental and processing files. After successful completion of this batch file, the user can directly go to the E&P Module. This process creates large files and the user should have at least 600 MB of free hard drive space available. Recall that if a full integrated run and/or the final processing by the Production Accounting Module is desired, this batch file should be run.

E. Description of Module Components and Programs

The Reservoir Performance Module is composed of a series of interrelated program loops, which calculate the summary economics and production response estimates for each individual reservoir on a sequential basis. The details of the program are described below in Figure 4.

Figure 4
Flow Chart for Reservoir Performance Module



E.1 Read in Data

Once the Reservoir Performance Module is invoked, GSAM selects the first reservoir and extracts data from the Resource Module output files (.GSM files) specified in REGIONS.DAT and determines whether the reservoir in question is a known reservoir or an undiscovered reservoir. This process of sequentially selecting and reading data from each reservoir is the outermost loop of the Reservoir Performance Module. Each reservoir undergoes calculations for several technology, price, and cost scenarios, the results of which are all stored before the next reservoir is analyzed.

Once read into the program, reservoir data enters the *technology loop*. This section of the Module examines the production effects of the implementation of various technology cases on each reservoir. GSAM can examine one or two technology cases in a single run (but normally considers two, current and advanced). The number of technologies is set and must be the same in the TECH.DAT and COST.DAT files. Some of the main parameters used to define each technology case include: the probability of drilling a development dry hole, the number of years after initial production of the reservoir in which infill drilling will be used on water-drive reservoirs, minimum system pressure for production, fracture length, fracture conductivity, and skin factors (all enhanced with advanced technology). This information is contained in TECH.DAT.

After reading the input parameters, the reservoir rock properties are distributed into three compartments or pay grades based on the resource type being analyzed. The purpose of these pay grades is to simulate non-homogenous nature of most reservoirs. Each pay grade has its individual characteristics as follows: pay grade 2 is specified as having the same reservoir properties as listed in the resource database; pay grade 1 contains the best portion of the reservoir; and pay grade 3 contains the worst portion of the reservoir. Data and methodologies used to distribute the pay grades are contained in the data file GEOLOGY.DAT. The factors in GEOLOGY.DAT are specified in such a manner that their multiplication and then addition for all the pay grades yield unity.

For known reservoirs with available historical data, history check calculations are performed. GSAM checks each reservoir's production values reported in 1997 (for Canada—1994) and total reserves produced through 1997 (for Canada—1994). 1997 is the year of the most recent set of complete NRG data for U.S. reservoirs and 1994 is the same year for Canada. If these two pieces of information are available, it means that the well has been produced in the past, and GSAM determines that a history check must be performed to account for past production and subsequent changes in reservoir properties and operational characteristics. If no past production is found, it is assumed that the reservoir is new and it "produces" without performing a history check (i.e., discovered undeveloped reservoirs). GSAM performs the history check calculations on known reservoirs by depleting the reservoir down to 1997 (1994 for Canadian reservoirs) reserve levels. The reservoir conditions (pressure, remaining reserves, etc.) obtained after these calculations form the initial conditions for GSAM to predict future production from the reservoir.

E.2 Type Curve Module

The next step of the Reservoir Performance Module calls the type curve module. This module can assess any of the six different reservoir type curves to predict production from the reservoir. The selection of type curve is based on the resource type defined in the GSAMID for undiscovered resource and by ‘Module1’ database element (specified in the .GSM file) for discovered producing resource. The six reservoir types that GSAM can analyze are: (1) conventional, (2) tight, (3) dual porosity (radial flow), (4) dual porosity (linear flow), (5) water drive, and (6) coal/shale reservoirs.

The type curve analysis yields gas production in annual time steps for three pay grades, three development options (primary, re-fractured, and infill), and two technology cases (current and advanced). Altogether, 18 separate analyses are performed for each reservoir (three pay grades, three development options, two technology scenarios). Among other things, the analysis computes the number of primary wells drilled in order to maintain reservoir production. Input and output format files for the type curve model (.TCI, .TCO files, etc.) can be generated by exercising the options in the REGIONS.DAT file. The files are prepared for individual reservoirs in files named using the GSAMID with DOS file name extensions .TCI (which stands for Type Curve Input) and .TCO (Type Curve Output). As these files are generated for each reservoir, they consume a good deal of space. The primary use of these files is in debugging, when only a few selected reservoirs are run through the Module to check for consistency in certain parameters.

In running the type curve model, binary (.BIN) files are created. These .BIN files store all of the reservoir performance data output from the type curve. If a subsequent run of the Reservoir Performance Module contains alterations in only economic (i.e., costing) data, in other words, the technology and geologic data remains constant, all of the data which would be generated by the type curve model is already stored in the binary files, and may simply be read from .BIN file. Bypassing the type curve process allows for significant reductions in the Module’s run time. However, if the user is unsure as to whether technology or geologic parameters have changed, or if this is the first run through the Reservoir Performance Module, the type curves should be employed. The type curve option is given in the REGIONS.DAT file.

E.3 Cash Flow Analysis

After the type curve process has been completed, the Reservoir Performance Module performs cash flow analyses. Costs are set based on information in COST.DAT, including drilling and completion, overhead, compression, and fixed and variable O&M. Specific tax rates used in the economic evaluations are set in TAXES.DAT and TAX_NAT.DAT for state and federal parameters. The purpose of this section is to calculate a range of production and operating costs that bound the potential costs under all reasonable potential operating scenarios. This is accomplished by running the following four cash flow analyses: (1) gas price of $2/Mcf, with all applicable costs considered; (2) gas price of $5/Mcf, with all applicable costs considered; (3) gas price of $2/Mcf, with drilling costs set to zero; and (4) gas price of $2/Mcf, with all operating, facility, environmental, etc. costs set to zero. Once the cash flow analysis is complete for each technology case, the Reservoir Performance Module places the output in the decision (.DEC) and production (.PRD) files to report the results.

F. *Frequently Changed Parameters*

The nature of the Reservoir Performance Module is such that not more than a few input parameters will most likely be altered. Two files, TECH.DAT and COST.DAT, allow for manipulation (remember, these files correspond to specific regions and technologies). If less than the two standard technology cases, current and advanced, are desired, this number must be set in both files, which are explained in detail in Appendix B. In the COST.DAT file, the drilling costs, operating costs, discount rate, and various cost factors can be changed. In TECH.DAT, factors such as the dry hole development well probability, the skin factor, horizontal well specifications, as well as many other technology parameters can be set. To investigate the sensitivity certain parameters from these files can be changed, such as, the entries in COST.DAT and TECH.DAT files and executing RESVPERF.EXE executable. As always, the user should store the results of the previous runs in a different directory (or previous files should be renamed) before the new run is executed.

VI. EXPLORATION AND PRODUCTION MODULE (3)

A. Summary Description of the Exploration and Production Module

The Exploration and Production (E&P) Module provides the logic for all investment decisions on upstream activities in GSAM. The module uses the summary decision files (.DEC) and production files (.PRD) from the Reservoir Performance Module, calculates minimum gas price requirements for each investment opportunity based on market factors adjusted over time, and ranks options consistent with an operator's approach to decision-making. Output includes summary information on production, reserves, and wells drilled either by region, by play, by resource type, or for the nation as a whole. Additional output files are created for use in the Demand and Integrating Module and the Production Accounting Module.

Exploration and production in GSAM can be evaluated under a variety of technology options. The parameters for assessing current and advanced technology penetration rates over time are contained in files ETEC_PEN.SPC and DTEC_PEN.SPC. Incremental technology penetration rates for federal lands are specified in DTEC_FED.SPC and ETEC_FED.SPC files. These files have information on the rate of technology penetration for each resource type being evaluated, including the percent of penetration into the market and its relative cost to undertake. Examples of the formats of the technology penetration files currently used in GSAM are included in Appendix C.

The Exploration and Production Module can be run in two modes. For estimating activity under a single established gas price track, the module is run one time with the price track file stored in some existing gas price file (GASPRC.NEW file copied from GASPRC.A99 for AEO price track or GASPRC.STR for starting guess for integrated run). For estimating upstream decisions and production responses leading to a market equilibrium in an integrated run, the Demand and Integrating Module will generate the GASPRC.NEW price track file and feed it back to the Exploration and Production Module.

B. Required Files

Input requirements for the Exploration and Production Module include the decision, production, and environmental files (DEC, .PRD, and. ENV files located in the \GSAM\EXPLPROD\TMP directory) from the Reservoir Performance Module and a series of E&P files (.SPC files) that define the starting market conditions and factors that change over time. In addition to that .ASM files (for vertical/horizontal selection purposes for undiscovered reservoirs only) are also copied from the Reservoir Performance Module and used in the selection procedure.

Before the E&P Module can be run, however, three steps need to be completed and checked to prepare the Reservoir Performance Module's outputs for the Exploration and Production Module. The first step involves using the undiscovered and undeveloped reservoirs which were run through the Reservoir Performance Module with horizontal drilling technology, then choosing either horizontal or vertical technology based on the reservoirs' investment efficiency economics. If the horizontal well option was not desired in the RP Module, then only the vertical well Reservoir Performance Module runs are used for the E&P Module. Although this vertical-horizontal well selection process could be seen as a Reservoir Performance function, it is placed in the E&P Module because the all of the RP processing has already been completed, and the output placed in the E&P directory. The second step creates binary "data bank" files, which compress the information stored in the .DEC, and .PRD files, to allow for significant reductions in run time and storage space. The third step involves creating environmental and processing files (using .ENV files from Reservoir Performance module) which contain environmental compliance and gas processing costs. The following files are necessary :

B.1 Base/Setup Files

B.1.1 Vertical/Horizontal Well Selection Routine (Can be skipped if only vertical wells are to be evaluated)

Name	Description	Location
F_SELECT.EXE	FORTRAN executable which selects the best vertical/horizontal drilling alternative for undiscovered and undeveloped reservoirs (for field size classes specified in FSCRESV.SPC file based on investment efficiency)	\EXPLPROD\TMP
SELECT.BAT	Batch file which will run the selection routine for one region and copy the final output as corresponding .PRD and .DEC files (users need to type "select undiscf" for performing selection on UNDISCF.* files, etc)	\EXPLPROD\TMP
SELALL.BAT	Batch file which will run the selection routine for all regions and copy the files in appropriate .DEC and .PRD files	\EXPLPROD\TMP
FSCRESV.SPC	Specifies filed size class definitions and the field size class for which horizontal well alternative can be selected	
RP Files (*.ASM, *.DEC, *.PRD) (See B.1.2 below for exact files)	Reservoir Performance Module files should be complete and stored in the correct location, output from the RP Module should be sent to \EXPLPROD\ TMP\VERTICAL and \EXPLPROD\TMP\ HORZ. RP batch files will put the proper files here	\EXPLPROD\TMP\ VERTICAL and \EXLPROD\TMP\HORZ

B.1.2 Files Needed for Creating “ Reservoir Banks”

Name	Description	Location
.PRD, .DEC Files	Output from Reservoir Performance Module (or from selection routine as explained in earlier step)	\EXPLPROD\TMP
UND*.GSM Files	Output from Resource Module, undiscovered only (such as UNDISCF.GSM, etc.)	\EXPLPROD\TMP
SPEC.DTD	Specifies the regions which will be in the discovered bank files, must be copied to SPEC.DAT when used	\EXPLPROD\NEWS
SPEC.DTU	Specifies the regions which will be in the undiscovered bank files, must be copied to SPEC.DAT when used	\EXPLPROD\NEWS
MAKEBIN.EXE	FORTRAN executable which creates bank files	\EXPLPROD
BINARY.BAT	Batch file used to create both the discovered and undiscovered reservoir bank files	\EXPLPROD

B.1.3 Environmental/Processing Cost Files

Name	Description	Location
SEQUEN.DAT	Specifies the order of .ENV files; NOTE: must match the order of files listed in SPEC.DTU and SPEC.DTD respectively	\EXPLPROD\NEWS
.ENV files	Output from Reservoir Performance Module, contain reservoir specific data by GSAMID (such as depth, impurity level, location, royalties etc.) The files are GSAM01.ENV, GSAM02.ENV, etc.	\EXPLPROD\TMP
XXENV.DOE files	Contain compliance costs for different states by year.	\EXPLPROD\ENV
ENV_WRTE.EXE	FORTRAN executable which creates the environmental and gas processing files	\EXPLPROD
ENV.BAT	DOS batch file used to create environmental (such as ENV_STAT.SPC and XXDOE.DOE) files and processing (ENV_PROC.SPC) files. Users need to type “ENV DOE” for creating environmental/processing files	\EXPLPROD

B.2 Data Input Files and Programs

Once the “data bank” and environmental files have been created, the Exploration and Production Module can be run with the following input files, which define the starting market conditions, and factors that change over time:

Name	Description	Location
GEN_TML.SPC	Time period specifications, discount rate, royalty incentive specifications, model start year, price of crude oil, etc.	\EXPLPROD
NODE.SPC	Supply and demand region names and indicators (disregard entries in data elements four and five for E&P)	\EXPLPROD
PLY_DFN.SPC	Play definitions, including play level royalty rates (used	\EXPLPROD

	only in RP module) , percent of play on federal land (not used by E&P or RP or PA modules), etc.	
DRL_CST.SPC	Drilling cost factors by region and depth (contains also exploration cost factors)	\EXPLPROD
DRL_RCP.SPC	Drilling footage capacity factors by region	\EXPLPROD
DRL_CAP.SPC	Factors affecting future drilling capacity and costs	\EXPLPROD
TAX_CDE.SPC	Tax specifications by year (currently modeled as royalty incentive start year)	\EXPLPROD
TAX_DET.SPC	Tax details for each specification (not used currently)	\EXPLPROD
ENV_DAT.SPC	Environmental file specifications by year	\EXPLPROD
ENV_STAT.SPC	Output from environmental file creation program, contains static reservoir information	\EXPLPROD
ENV_PROC.SPC	Gas processing cost by GSAMID	\EXPLPROD
EXP_CST.SPC	Exploration cost data by play and technology (not used currently)	\EXPLPROD
DVL_TPR.SPC	Development technology data	\EXPLPROD
ETEC_PEN.SPC	Exploration technology penetration rate	\EXPLPROD
DTEC_PEN.SPC	Development technology penetration rate and relative costs	\EXPLPROD
EXP_DFN.SPC	Specification for Exploration Risk by Resource	\EXPLPROD
EXP_PLY.SPC	Specification for Exploration Risk by Play	\EXPLPROD
GASPRC.NEW	Price file used in the analysis (passed from Integrating Module or set independently). File contains historical gas prices for the period 1993-1996 and forecasts from 1997 onwards	\EXPLPROD
UNDBNK.SPC	Undiscovered resource multiplier and Undiscovered resource factors by FSC and reserve growth for each field size class within each GSAM region	\EXPLPROD
89RESAV.SPC	Controls undiscovered reserve availability in relation with effective penetration rates of exploration drilling	\EXPLPROD
SUP_CSE.DTX	Specifies the price track number in the gas price (GASPRC.NEW) file (1-5)	\EXPLPROD
PRO_DAT.SPC	File used in creating gas processing costs	\EXPLPROD
PRO_REG.SPC	File used in creating gas processing costs	\EXPLPROD
PRO_IMP.SPC	File used in creating gas processing costs	\EXPLPROD
RESAVRG.SPC	Annual reserve growth rate by GSAM region as specified by USGS	\EXPLPROD
BASE.BAT	Batch file to run E&P module at predefined base case.	\EXPLPROD
AEO.BAT	Batch file to run AEO case	\EXPLPROD
EXPLPROD.EXE	FORTRAN executable file for the Exploration and Production Module	\EXPLPROD
ETEC_FED.SPC	Contains Federal Lands Technology Penetration Increments	\EXPLPROD

	for Exploration Technology	
DTEC_FED.SPC	Contains Federal Lands Technology Penetration Increments for Development Technology	\EXPLPROD
SUPPLY.HIS	Contains historical data for the wellhead gas price and the supply of natural gas by year (1993-1996) and GSAM region	\EXPLPROD
PRODEXPL.BAT	Batch file to run Exploration and Production Module with three parameters (case (such as BASE.BAT for base case), gas price file, gas price track)	\EXPLPROD

It should be noted that the user may create or alter any of these unique data input file, so long as the formatting is the same as the original input file (the E&P Module reads primarily fixed-format data, so that the placement and decimals must be consistent), the DOS file name is the same as the original (with a different extension), and the file to be used is copied so that its extension is .SPC. Samples of input files currently used in the Exploration and Production Module are provided in Appendix C, with detailed explanations.

C. Output Files

The Exploration and Production Module will create the following output files:

Name	Description	Location
PRODSUMM.OUT	Detailed listing of production, OGIP of reserves, wells drilled, operating wells, etc. by region or play	\EXPLPROD
SUPPSUMM.OUT	Supply volumes by region/resource type	\EXPLPROD
REVSUMM.OUT	Summary of reserves by region	\EXPLPROD
PRICE.OUT	Gas price at which the Module was run, as well as other information, such as exploration wells drilled, average depth drilled by year (used in production accounting module) and drilling cost factors	\EXPLPROD
DECISION.OUT	Summary of investment decisions (used in Production Accounting Module)	\EXPLPROD
SUPPLY.EXT	Interim supply values (used in demand and integrating module)	\EXPLPROD
WELLSUMM.OUT	Contains summary for the number of wells drilled by region, well type and resource type	\EXPLPROD
NRRSUMM.OUT	Contains summary data for the Number of Reservoir Accumulations by year (1997-2020), FSC, region and technology	\EXPLPROD
RGRRSUMM.OUT	Contains summary data for the Reserve Growth of Recoverable Reserves by year (1997-2020), FSC, region and technology	\EXPLPROD

BNRRSUMM.OUT	Contains summary data for the Banked (i.e. Discovered/Undeveloped) Recoverable Reserves by year (1997-2020), FSC, and region	\EXPLPROD
UNRRSUMM.OUT	Contains summary data for the Undiscovered Recoverable Reserves by year (1997-2020), FSC, region and technology	\EXPLPROD
EXPLWLS.OUT	Contains summary for the number of exploration wells drilled by play for the period 1997-2020	\EXPLPROD

D. Operational Procedures for Running the Exploration and Production Module

In the following sections two options for running the Exploration and Production Module: (1) manual operation and (2) automation through batch files are explained in detail. The DOS batch files will combine many of the manual steps, and will produce exactly the same results as the manual operation, if both are run correctly. Manual operation should be used in testing (for example, to experiment with some changed data parameter) or to identify exactly where problems occur if the model is not running. Batch files should be used in most other cases, as performing a complete run using the manual steps would be a repetitive, time-consuming, and error-prone activity. The Module can be run as a stand-alone component or as a piece of a fully integrated GSAM run in conjunction with the Demand and Integrating Module. Instructions for both options are given below.

Before the Exploration and Production Module can be run the user must perform four sets of procedures. They are:

1. Decide whether to run the dataset of horizontal or vertical wells; if horizontal well technology is to be an option, the selection routine must be run

2. Create "bank" files

3. Create Environmental files

4. Ensure all files are stored in generic .spc filenames and the file GASPRC.A99 is copied to GASPRC.NEW (for AEO 1999 run). Instructions for the creation of these preliminary files are given below:

D.1 Manual Operation

D.1.1 Horizontal vs. Vertical Well Selection (Optional, May Be Skipped If Desired)

D.1.1.1 When running only the horizontal wells dataset:

a) Check whether all the .ASM, .PRD, .DEC, and .ENV for the horizontal wells are in the \EXPLPROD\TMP\HORZ directory
b) Copy them to the \EXPLPROD\TMP directory

D.1.1.2 When running only the vertical wells dataset:

a) Check whether all the .ASM, .PRD, .DEC, and .ENV for the vertical wells are in the \EXPLPROD\TMP\VERTICAL directory
b) Copy them to the \EXPLPROD\TMP directory

D.1.1.3 When running a mixed dataset with both vertical and horizontal wells:

NOTE: If a full run is to be submitted to the Production Accounting Module, the selection program must be run for vertical and horizontal wells (SELALL.BAT for all regions or SELECT.BAT for one file at a time, see section B.1.1 for running procedure)

NOTE: If a batch file was used in RP Module and the directory structure is correct, the necessary files will already have been copied to the correct directories

a) Check whether all the .PRD, .DEC, .ASM, and .ENV files for the vertical and horizontal wells are in the \EXPLPROD\TMP\VERTICAL and \EXPLPROD\TMP\HORZ directories respectively

b) Run SELECT.BAT in the \EXPLPROD\TMP directory by typing SELECT followed by the prefix of the name of the file with the desired dataset, for example run the dataset for undiscovered Canadian reservoirs by typing: SELECT UNDCAN,

c) Repeat step (a) until all necessary undiscovered and undeveloped files are accounted for with horizontal technology and costs

d) This selection process will generate a corresponding .OUT, .DEC, and .PRD file, which will reside in the \TMP directory of the E&P Module. Once all of the undiscovered and undeveloped reservoirs have been run through the selection process, all of the .OUT files are concatenated into VERHOR.GSM, a complete list of reservoir developed using either vertical or horizontal wells, which is used later in the Production Accounting Module. To create this file, run VERHOR.BAT (from \GSAM\EXPLPROD\TMP directory). VERHOR.GSM should then be copied to \PRODACCT\DATA for its use in the Production Accounting Module (If SELALL.BAT file is run, this procedure is automatically performed).

e) Copy .ENV files (such as GSAM01.ENV, GSAM02.ENV, etc.) from GSAM\EXEPLPROD\TMP\VERTICAL into GSAM\EXPLPROD\TMP\ directory.

D.1.2 Creating "Bank" Files

The bank files are created to allow for reductions in run time and storage space. The process uses the .PRD and .DEC files from the Reservoir Performance Module output (either with or without the selection program described above) and the undiscovered reservoir .GSM files from the Resource Module. These files should all be located in the \EXPLPROD\TMP directory.

a) Go to \EXPLPROD\NEWS directory
b) Choose the discovered regions to be used by altering SPEC.DTD or do the same for the undiscovered regions in SPEC.DTU file.

NOTE: Any or all regions may be chosen to be included. As long as the region was run through the RP Module, it is included by explicitly naming the .DEC and .PRD files in SPEC.DTD or SPEC.DTU (which has the .GSM files as well for undiscovered reservoirs).

NOTE: If the Exploration and Production Module is going to be utilized, there must be at least one undiscovered region (in SPEC.DTU). GSAM is an exploration-based model, and if there is no undiscovered resource, there is no exploration and the model does not run. Also, if an integrated run will be performed, it requires that all regions be in the 'bank".

c) Copy \NEWS\SPEC.DTD to \NEWS\SPEC.DAT
d) Run MAKEBIN.EXE from EXPLPROD directory (this will make the discovered "bank" files: DISB.BNK and DISB.TCP which will be created the EXPLPROD\NEWS directory)
e) Copy \NEWS\SPEC.DTU to \NEWS\SPEC.DAT
f) Run MAKEBIN.EXE (this will make the undiscovered "bank" files: UNDB.BNK and UNDB.TCP which will be created in the EXPLPROD\NEWS directory)

D.1.3 Creating Environmental Files

a) Verify that the order of files listed in \NEWS\SEQUEN.DAT matches exactly the order in \NEWS\SPEC.DTU followed by \NEWS\SPEC.DTD
b) Make certain that all .ENV files created from the Reservoir Performance Module are in the \EXPLPROD\TMP directory.
c) Copy a XXENV.%1 file (from \EXPLPROD\RNV directory to EXPLPROD\STATEREG.ENV
d) Run ENV_WRTE.EXE
e) Copy ENV3.OUT to XXDOE.%1, keeping the correct year in the DOS file name and keeping the extension naming convention consistent
f) Repeat steps c - e for all XXENV.DOE files.
g) Copy ENV_STAT.OUT to ENV_STAT.SPC and ENV_PROC.OUT to ENV_PROC.SPC

D.1.4 The Exploration and Production Module in a Stand-Alone Run

a) Copy GASPRC.XXX to GASPRC.NEW, depending on the desired gas price file

b) Choose a gas price track (1 through 5) and copy the corresponding SUP_CSE.DTX file to SUP_CSE.DAT. Make sure that the track number exists in the GASPRC.NEW file

c) Copy ENV_PROC.OUT to ENV_PROC.SPC and ENV_STAT.OUT to ENV_STAT.SPC (if not done in and earlier step)

d) Modify ENV_DAT.SPC so that it has the proper names of the XXDOE.XXX environmental files created form running the Environmental Module ENV_WRTE.EXE (this should involve changing only the extension names in the file)

e) Run EXPLPROD.EXE

D.2 Batch File Operation

The DOS batch file used to run the selection program on one region is \TMP\SELECT.BAT. It has one argument: the region name. Typing SELECT UNDCAN will run the undiscovered conventional Canadian file through the selection routine. \TMP\SELALL.BAT will run the selection program on all undiscovered and undeveloped regions and create VERHOR.GSM and copy the file in \GSAM\PRODACCT\DATA directory. Running the \EXPLPROD\BINARY.BAT file will create the undiscovered and discovered bank files, as long as SPEC.DTU and SPEC.DTD contain the desired files. ENV.BAT will create the environmental files, as long as the files listed in SEQUEN.DAT match those in SPEC.DTU and then SPEC.DTD and follow the same order, and the .ENV files from the Reservoir Performance Module and the XXENV.DOE files are in proper place.

The batch file that will run the Exploration and Production Module is named PRODEXPL.BAT. It has three arguments, which account for case (BASE.BAT for a base case run), price file, and price track. The first argument calls a batch file, which specifies the case being run (such as BASE.BAT, etc.), the second argument specifies the gas price file extension (such as gasprc.A99), and the third names the gas price track (1 through 5). For example, typing PRODEXPL ADV A99 1 would run the Module under the base case, with the AEO 1999 gas price file track (GASPRC.A99, see section E below for a description), and using the first price track. Any other scenarios may be created and incorporated, as long as another batch file is created (similar to the BASE.BAT file), which shares the same DOS file extension as that specified in the batch file's first argument (i.e. the case argument).

In an integrated run, batch files in the Demand and Integrating Module will run the Exploration and Production Module. Operation of an integrated run is described in detail in the next chapter.

E. Description of Module Components

The following gives a detailed description of each phase of the Exploration and Production Module, in the order in which they are performed, beginning with the creation of the preliminary files.

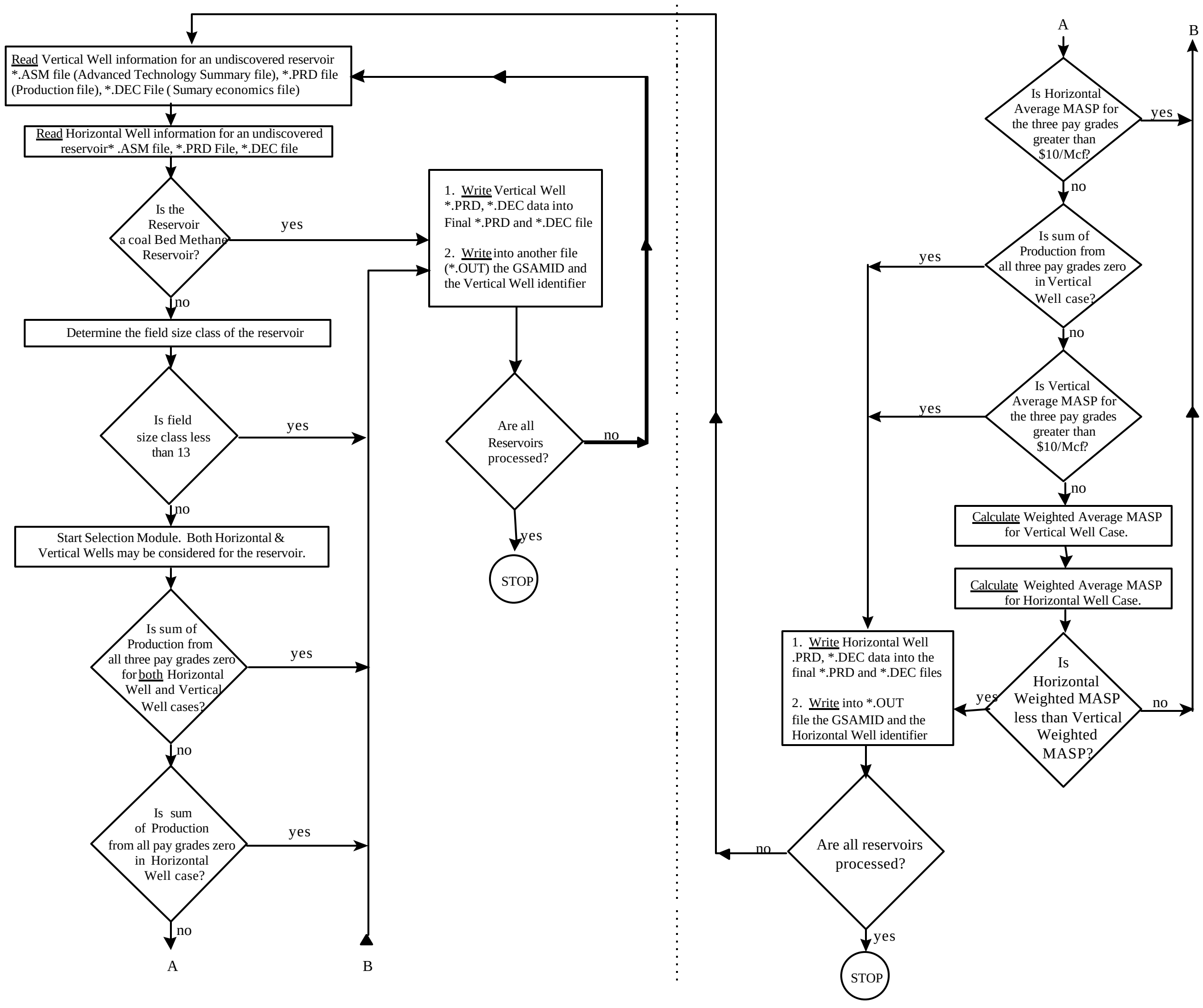
E.1 Horizontal versus Vertical Well Selection

GSAM is capable of analyzing undiscovered and undeveloped reservoirs using both horizontal and vertical well options. The horizontal well length is specified in TECH.HOR and costs are specified in COST.HCN/HUS. The skin factors are adjusted in the Reservoir Performance Module for horizontal wells from the entries of the corresponding vertical well entries.

The responses from horizontal wells and vertical wells are compared to each other and the better alternative (based on investment efficiency calculations for advanced technology) is selected. Investment efficiency (or present value ratio) is defined as the net present value of a project divided by the present worth of net costs. This selection is achieved by either the SELECT.BAT file or the SELALL.BAT batch file and is explained above and in Figure 5. It should be noted that any two different technology types (such as horizontal/vertical; horizontal/lower skin vertical, etc.) can be used in the selection process.

After the selection is done, the final .PRD and .DEC files are located in GSAM\EXPLROD\TMP subdirectory. The selection process (SELALL.BAT file) creates another file called VERHOR.GSM which contains the GSAMID and a horizontal/vertical well identifier which is used in the Production Accounting Module for costing purposes.

Figure 5
Horizontal vs. Vertical Wells: Selection Process in GSAM


Read Vertical Well information for an undiscovered reservoir *.ASM file (Advanced Technology Summary file), *.PRD file (Production file), *.DEC File (Sumary economics file)
Read Horizontal Well information for an undiscovered reservoir* .ASM file, *.PRD File, *.DEC file
Is the Reservoir a coal Bed Methane Reservoir?
yes
no
Determine the field size class of the reservoir
Is field size class less than 13
Start Selection Module. Both Horizontal & Vertical Wells may be considered for the reservoir.
Is sum of Production from all three pay grades zero for both Horizontal Well and Vertical Well cases?
Is sum of Production from all pay grades zero in Horizontal Well case?
A
B
1. Write Vertical Well *.PRD, *.DEC data into Final *.PRD and *.DEC file
2. Write into another file (*.OUT) the GSAMID and the Vertical Well identifier
Are all Reservoirs processed?
STOP
Is Horizontal Average MASP for the three pay grades greater than $10/Mcf?
Is sum of Production from all three pay grades zero in Vertical Well case?
Is Vertical Average MASP for the three pay grades greater than $10/Mcf?
Calculate Weighted Average MASP for Vertical Well Case.
Calculate Weighted Average MASP for Horizontal Well Case.
Is Horizontal Weighted MASP less than Vertical Weighted MASP?
1. Write Horizontal Well .PRD, *.DEC data into the final *.PRD and *.DEC files
2. Write into *.OUT file the GSAMID and the Horizontal Well identifier
Are all reservoirs processed?
STOP

E.2 Environmental and Processing Cost File Creation

GSAM is capable of taking environmental compliance costs on a state level. The environmental files contain the following information by state:

a) Tangible capital costs for existing wells
b) Intangible capital costs for existing wells
c) Operating and maintenance costs for existing wells
d) Tangible capital costs for new wells
e) Intangible capital costs for new wells
f) Operating and maintenance costs for new wells.

The ENV_WRTE program first reads the .ENV files (located in \EXPLPROD\TMP such as GSAM01.ENV etc.), as specified in the SEQUEN.DAT file. The .ENV file contains GSAMID, area, royalty rates, impurities levels (CO2, H2S, N2), and condensate yield. Then the reservoir is matched with the state-specific environmental costs from the XXENV.DOE files (located in \EXPLPROD\ENV directory). The output files, named XXDOE.DOE, are written in sparse format with only non-zero entries written in this final environmental file. The batch file names this file with the appropriate extension.

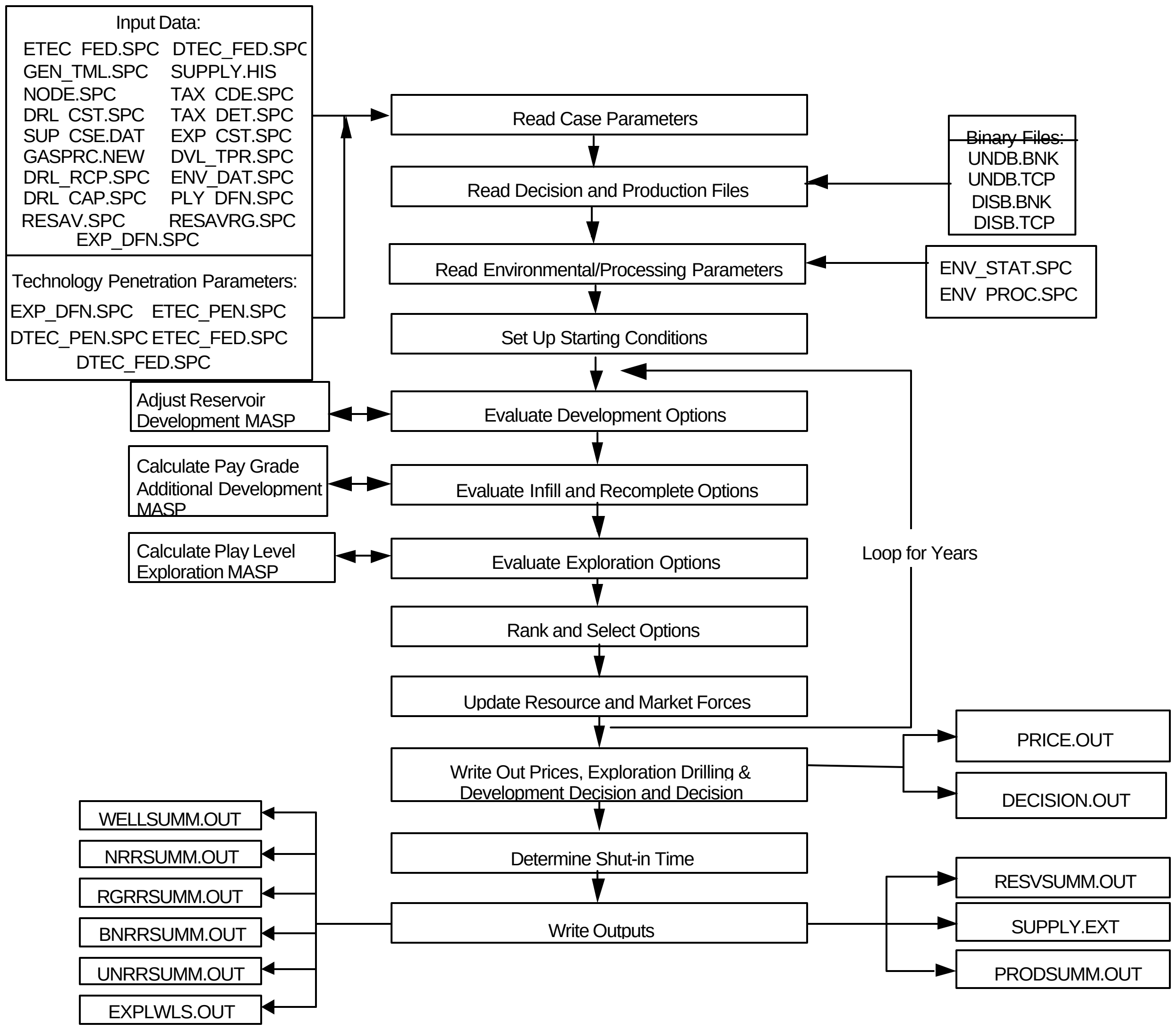
In addition to these environmental files which could change for different regulations, another file, ENV_STAT.OUT, is also created (it sequentially contains all the reservoirs to be evaluated in the E&P model in one file). It contains all the static information about reservoirs. This static file contains non-changing information about the reservoir and contains all the information as specified in the .ENV files listed in SEQUEN.DAT.

The ENV_WRTE program also creates processing costs for each reservoir. The .ENV file provides all the information needed for processing cost estimation. Processing costs in $/Mcf are calculated and stored in ENV_PROC.OUT file. Note that the files ENV_STAT.OUT and ENV_PROC.OUT have to be copied into ENV_STAT.SPC and ENV_PROC.SPC respectively so that the Exploration and Production Module can read them. The specification of the final environmental files (such as 98DOE.DOE, 99DOE.DOE, 00DOE.DOE, etc.) should be done in the ENV_DAT.SPC file for Exploration and Production Module processing.

E.3 The Exploration and Production Module

The Exploration and Production Module consists of several segments that are called as subroutines from the main program. These components provide the basis for making all required investment decisions over time. The price scenario and the technology parameters, as well as other input data, are factors in the decision process. Figure 6 shows the data and logic flow for the components of the Exploration and Production Module.

Figure 6
Flow Chart for Exploration and Production Module



The Module first reads input concerning the run and summary reservoir information from the Reservoir Performance Module.

Exploration technology is phased in through data in the ETEC_PEN.SPC file. Additionally, exploration technology parameters on dry hole factors and the relative capability of prospectors to find bigger accumulations are also modeled. These factors are stored in EXP_DFN.SPC for each resource. The model uses the factors for the resource type and field size class (from EXP_DFN.SPC).

Economic modeling in the Exploration and Production Module uses the summary results in the .DEC files to calculate the future profitability of exploration and development of each reservoir. This is completed as the market conditions and gas prices are adjusted over time. On each annual pass, reservoirs available for development, additional development, and exploration are evaluated, and their drilling, non-drilling, and tax costs are adjusted to contemporary conditions. The updated minimum acceptable supply price (MASP) is calculated and appropriately risked based on dry hole and exploration success factors to determine project economics. Once flagged, the reservoir is packed away until it is needed in the output phase.

The production and reserves summary report is output and named PRODSUMM.OUT. This file displays information on OGIP of reserves, production, wells drilled, and annual gas prices for the full United States, Canada, each GSAM supply region, and each resource type. The Module also produces RESVSUMM.OUT, in which summary reports on resource and reserves developed in each play, region, resource type, or country are contained. This report provides information on original gas in place (OGIP) available for discovery, discovered during the analysis, and developed during the analysis, and reserves estimates for total, primary, and secondary (from infill-drilling or recompletions) operations. A supply summary file SUPPSUMM.OUT is also written which contains total gas produced by region and by resource type. Other files such as RGRRSUMM.OUT, BNRRSUMM.OUT, UNRRSUMM.OUT, NRRSUMM.OUT, EXPLWLS.OUT, WELLSUMM.OUT are also created.

Key output files also include DECISION.OUT and PRICE.OUT, (decision and price files, respectively) which are inputs to the Production Accounting Module. These files contain information on all investment decisions made by the E&P Module, as well as the regional gas price and drilling factors used in the analysis. This information is provided for individual GSAM regions over the entire time-frame of the run. A supply response file is created as each gas price is run through the Exploration and Production Module. The results are stored in SUPPLY.EXT. This file, combined with other price cases evaluated, is converted to SUPPLY.SPC, which forms the supply curve for the Demand and Integrating Module of GSAM.

A regional gas price file has been created for analysis. GASPRC.A99 contains DOE/EIA Annual Energy Outlook (AEO) 1999 regional price estimates. The first model year is 1997. It also contains historical gas prices for years 1993, 1994, 1995, and 1996, which are kept there to understand historical gas price trends. The price tracks of this file include the AEO Reference Case and use the batch file AEO.BAT to run the E&P module with this gas price file.

F. Frequently Changed Parameters

In the Exploration and Production Module, several input files have parameters, which are often changed for alternative analyses. ETEC_PEN.SPC and DTEC_PEN.SPC can control the rate of technology penetration by year. Remember that these file names are the generic names and that they are case-specific (one exists each for current and advanced technology).

The number of years for analysis, as well as the start year, can be set in GEN_TML.SPC. GEN_TML.SPC also contains easily modified information on the discount rate and on the royalty incentive parameters. Also, the gas price file is often modified. This can only be done in a stand-alone Exploration and Production run, as an integrated run generates gas prices. Recall that the name of the gas price file must be copied to GASPRC.NEW and that a stand-alone run is price track-specific.

The input files DRL_CST.SPC and DRL_CAP.SPC contain some frequently altered parameters. The exploration drilling cost factors by region are listed in DRL_CST.SPC. After the region number are four data elements, which contain drilling, cost equation coefficients. The fifth data element contains a factor, which describes how much more costly drilling exploration wells is compared to drilling development wells. Note that only exploration drilling costs are calculated and factored into the Exploration and Production Module. Development drilling costs are calculated in the Reservoir Performance Module, with the file COST.DAT containing the drilling cost factors. The file DRL_CAP.SPC contains easily altered data on drilling efficiency (of development wells compared to exploration wells), annual exploratory drilling cost reductions (costs decline a certain percentage per year due to advancing technology), and other exploratory drilling parameters. For more detail on the Exploration and Production Module data input files, refer to Appendix C.

VII. DEMAND AND INTEGRATING MODULES (4)

A. Summary Description of the Demand and Integrating Modules

The Demand and Integrating Modules consist of several related computer models that are run in sequence. The calculation of demand for natural gas in the residential, commercial, industrial, and electrical power generation sectors is done in the Demand Module. These demand values are then equilibrated with supply quantities from the E&P Module to arrive at a balanced market. This is the function of the Integrating Module, which accomplishes this by solving an associated linear program (LP) while also considering pipeline flows, storage and peak-shaving usage. The Integrating Module returns a file (GASPRC.NEW) which is used directly by the Exploration and Production Module to calculate an estimate of equilibrium supplies based on the updated regional wellhead gas prices.

B. Required Files

The Demand and Integrating Modules provide estimates of regional demand by sector, season, and year and provide the logic to balance supply and demand in order to minimize the overall cost of gas utilization.[1] The inputs are contained in the following files:

B.1 Data Input Files

Name	Description	Location
GEN_TML.SPC	Model start, final, and current year, days in the seasons, discount rate	\DEMDINTG
NODE.SPC	Supply and demand region names and indicators.	\DEMDINTG
COAL_PR.SPC	Coal prices by region and time period (in $/Mcf equivalent)	\DEMDINTG
LINK_NDE.SPC	Capacity, costs, fuel requirements, and expansion factors for pipelines	\DEMDINTG
DMN_SEC.SPC	Mark-up factors by region for each sector	\DEMDINTG
OTH_SUP.SPC	Other supply specifications (LNG, ANGST, other supply projects with unique economic factors and constraints)	\DEMDINTG
RES_DEM.SPC	Base residential demand specifications	\DEMDINTG
COM_DEM.SPC	Base commercial demand specifications	\DEMDINTG
EU_DEM.SPC	Base electric utility demand specifications	\DEMDINTG

[1] The actual objective function concerns the optimization of the consumer surplus plus producer surplus less transportation and investment costs, discounted over time.

EU_GEN.SPC	Electric utility cost and efficiency data by plant type	\DEMDINTG
GASPRC.STR	Starting gas price file, copied to \EXPLPROD\ GASPRC.NEW in a full integrated run	\DEMDINTG
CTGPRC.DAT	Citygate (wholesale) price file	\DEMDINTG
CTGPRC.STR	Starting citygate (wholesale) price file	\DEMDINTG
GASALL.BSS	Advanced starting basis for the Integrating Module's LP (OPTIONAL)	\DEMDINTG
SROM.IN	Specifies properties of individual storage reservoirs, output from the Storage Reservoir Performance Module	\DEMDINTG
PFLAG.SPC	Flag for printing certain intermediate outputs (1 = print, 0 = do not print)	\DEMDINTG
BARRIER.CFG	LP solver configuration file	\DEMDINTG
DISTIND.SPC	Regional markups for distillate crude in the industrial sector	\DEMDINTG
DISTELE.SPC	Regional markups for distillate crude in the electricity generation sector	\DEMDINTG
1PCTELE.SPC	Regional markups for 1% sulfur residual crude in the electricity generation sector	\DEMDINTG
1PCTIND.SPC	Regional markups for 1% sulfur residual crude in the industrial sector	\DEMDINTG
3PCTELE.SPC	Regional markups for 3% sulfur residual crude in the electricity generation sector	\DEMDINTG
3PCTIND.SPC	Regional markups for 3% sulfur residual crude in the industrial sector	\DEMDINTG
REFMARG.SPC	Gulf Coast refinery margins for crude (by type of crude)	\DEMDINTG
BOILERS.SPC	Data for the US industrial boilers in the module	\DEMDINTG
CANBOIL.SPC	Data for Canadian and Mexican industrial boilers by group	\DEMDINTG
CANNUGS.SPC	Data for Canadian and Mexican industrial NUGs by group	\DEMDINTG
CAPS.SPC	Forced NEW (not cumulative) capacity file by node	\DEMDINTG
COM_EFF.SPC	Commercial energy efficiency by GSAM Demand region	\DEMDINTG
COM_ELS.SPC	Commercial demand elasticities by GSAM demand region	\DEMDINTG
COM_GNP.SPC	Commercial sector GNP values and growth rates by GSAM demand region	\DEMDINTG
COM_LD.SPC	Seasonal commercial load factors by GSAM demand region	\DEMDINTG
COM_PRC.SPC	Commercial gas prices and growth rates by GSAM demand region	\DEMDINTG
CRUDE.SPC	History and forecasts for crude oil prices	\DEMDINTG
DMNCRV.SPC	Parameters for the step function approximations of the sectoral demand curves	\DEMDINTG
DUAL_PRC.SPC	Dual prices by node, year and load season	\DEMDINTG
DUAL_PRC.STR	User-specified starting dual prices by node, year and load season	\DEMDINTG
EFF.SPC	Base energy efficiency by GSAM demand region	\DEMDINTG

EU#_LD.SPC	Load factors by electric utility seasons, gas seasons. and GSAM demand regions (“#” denotes the electric utility season and ranges from 1 to 4)	\DEMDINTG
FEED.SPC	Industrial feedstock demand for natural gas	\DEMDINTG
FLOWS.SPC	Forced pipeline flows among the nodes of the module	\DEMDINTG
GASPRC.HIS	Historical gas prices from 1993 to 1997 by GSAM supply region and price track	\DEMDINTG
IND_EI.SPC	Energy intensity of the industrial boilers and NUGs	\DEMDINTG
IND_ELS.SPC	Industrial elasticities of the different fuels by GSAM demand region	\DEMDINTG
IND_LD.SPC	Industrial load factors by season and GSAM demand region	\DEMDINTG
IND_PRC.SPC	Industrial base gas prices by fuel and GSAM demand region	\DEMDINTG
LNG.SPC	LNG capacities and costs for existing and new plants	\DEMDINTG
NOX.SPC	NOx allowance costs by fuel, GSAM demand region, start year, and gas season	\DEMDINTG
NUGS.SPC	Parameters for the NUGs (non-utility generators) in the Industrial demand module	\DEMDINTG
ODUALS.SPC	Dual gas prices from each previous iteration of the model by GSAM demand region and year	\DEMDINTG
ODUALS.STR	User-specified starting dual gas prices by GSAM demand region and year	\DEMDINTG
POP_GRP.SPC	Gross Regional Product and Population by GSAM demand region, and Total Electricity Demand for all demand regions	\DEMDINTG
PROC.SPC	Industrial process heat demand for gas by GSAM demand region	\DEMDINTG
PROPANE.SPC	Propane/air capacities and costs by GSAM demand region	\DEMDINTG
RES_EFF.SPC	Residential energy efficiency by GSAM demand region	\DEMDINTG
RES_ELS.SPC	Residential demand elasticities by GSAM demand region	\DEMDINTG
RES_LD.SPC	Residential seasonal load factors by GSAM demand region	\DEMDINTG
RES_POP.SPC	Residential sector population by GSAM demand region	\DEMDINTG
RES_PRC.SPC	Residential gas prices by GSAM demand region	\DEMDINTG
SOX.SPC	SOx allowance costs by fuel, GSAM demand region, start year, and gas season	\DEMDINTG
STORVALS.SPC	Decline percentage for extraction rate from storage reservoirs and percentage of storage capacity to be used by the model	\DEMDINTG
SUP_LD.SPC	Supply load factors by season and GSAM supply region	\DEMDINTG
SUPPLY.SPC	Supply prices and quantities by GSAM supply region	\DEMDINTG
WEATHER.SPC	Seasonal weather-related adjustment parameters for the gas demand by GSAM demand region and year (1993-2020).	\DEMDINTG

These data files are used to provide a starting, baseline demand for natural gas in each region and sector. Future gas demand is evaluated based on these input parameters, as described below.

Samples of each of these files are provided in Appendix D. It should be noted that the user may create any unique data input file, so long as the formatting is the same in that in the original input file, the DOS file name is the same as the original (with a different extension), and the file to be used is copied so that its extension is .SPC, .DAT, IN, or .STR (as is appropriate).

B.2 Program Files\DEMDINTG

Name	Description	Location
RUNGSAM.BAT	DOS batch file which runs one integrated run	\DEMDINTG
TCOMB.BAT	DOS batch file which prepares .MPS files for the LP solver	\DEMDINTG
INTMGN.EXE	Executable that generates data for the integrating LP	\DEMDINTG
MP-OPT.EXE	Executable which solves the LP (3rd party software)	
INTRPT.EXE	Executable which writes reports	\DEMDINTG
INTRVS.EXE	Executable which outputs GASPRC.NEW and checks for convergence of the equilibrium process	\DEMDINTG
DASH10.BAT	DOS batch file that runs the full ten passes in an integrated run.	\DEMDINTG
DASHB.BAT	DOS batch file which runs one pass in an integrated run	\DEMDINTG
B.BAT	Invokes the LP solver software (Newton-Barrier method)	\DEMDINTG

C. Output Files

The Demand and Integrating Modules create the following output files:

Name	Description	Location
E&P Module Outputs	Outputs from the Exploration and Production Module (see previous section)	\EXPLPROD
GSAMSLN.RPT	Transportation flows and material balance for each region	\DEMDINTG
GSAMSLN.STA	Summary of storage activity	\DEMDINTG
GSAMSLN.STC	Summary of storage cost	\DEMDINTG
GSAMSLN.FLE	Report on supply and demand estimates, detailed electrical power sector report	\DEMDINTG
GSAMSLN.SEA	Report on seasonal output by GSAM region, sector, season, and year	\DEMDINTG
GSAMSLN.SUP	Report on supply sources by load period, GSAM region and year	\DEMDINTG
GASPRC.NEW	Output gas price estimates, can be sent back to E&P Module to generate a new supply curve	\DEMDINTG
GASALL.BSS	Basis file, can be useful in subsequent runs for advanced starting points	\DEMDINTG
GASALL.PRT	Solution from the LP software (ASCII format)	\DEMDINTG
GASALL.SOL	Solution from the LP software (non-ASCII format)	\DEMDINTG

D. Operational Procedure for Running the Demand and Integrating Modules

The Demand and Integrating Modules calculate market equilibrium gas prices of quantities by utilizing the Exploration and Production Module to generate a supply curve, the Demand Module to generate demand estimates, and the Integrating Module to balance supply and demand. To generate a supply curve, the Exploration and Production Module is run four successive times, each with a different price track, in essence generating four points on the supply curve for each region-year combination. The resulting file, SUPPLY.SPC, is sent to the Integrating Module where supply and demand are equilibrated. The Integrating Module in turn computes estimates of market equilibrium gas prices, producing the file GASPRC.NEW that is sent to the Exploration and Production Module to generate the corresponding supply levels. This process is carried through several iterations (usually 10) so that an equilibrium in gas prices and quantities will be computed.

The following describes how to run one pass of the Demand and Integrating Modules manually. However, because it takes several passes between the E&P Module and the Integrating Module before price converges to an equilibrium level, seldom will only one pass be desired, so for a multiple-iteration integrated run, use the directions for the DOS batch file, which are below.

D.1 Manual Operation

NOTE: Because the Demand and Integrating Modules estimate a market equilibrium by calculating regional prices taking into account transportation constraints, all regions must be used. Before running, ensure that all regions have been run through the Reservoir Performance Module and that the reservoir-level data is put into "bank" files.

a) Confirm that time period specifications in the GEN_TML.SPC file in Demand and Integrating directory and GEN_TML.SPC file in Exploration and Production Module are the same. Also, check the royalty incentive parameters in the GEN_TML.SPC file of the E&P Module.

b) Confirm that the Exploration and Production Module is configured as desired (especially ETEC_PEN.SPC and DTEC_PEN.SPC).

c) Copy GASPRC.STR to \EXPLPROD\GASPRC.NEW (This provides a starting gas price file for the E&P Module).

d) Copy CTGPRC.STR to CTGPRC.DAT (This gives a starting point for the citygate prices (wholesale prices by region).

e) Run the Exploration and Production Module four times with price tracks 1,2,3, and 4 and save each SUPPLY.EXT as SUPPLY.EX1, SUPPLY.EX2, SUPPLY.EX3, and SUPPLY.EX4, respectively.

f) Create SUPPLY.SPC by combining SUPPLY.EX1 through SUPPLY.EX4 into 1 file, and copy SUPPLY.SPC to \DEMDINTG.

g) In the \DEMDINTG directory run INTMGN.EXE then TCOMB.BAT.

h) Initiate the linear program solver by running B.BAT.

i) Run INTRPT.EXE then INTRVS.EXE to check for convergence, create reports, and format the modules' output.

D.2 Batch File Operation

The Industrial Demand and Integrating Modules, in sequence with the Exploration and Production Module, are run a series of times to compute estimates of market equilibrium prices and quantities. This is done by executing a DOS batch file named RUNGSAM.BAT, which has one argument. For example, by typing RUNGSAM A, the E&P, Demand, and Integrating Modules will generate results using extensions A01 through A10 (for each of the ten iterations), which will be saved in files with extension A01 through A10.

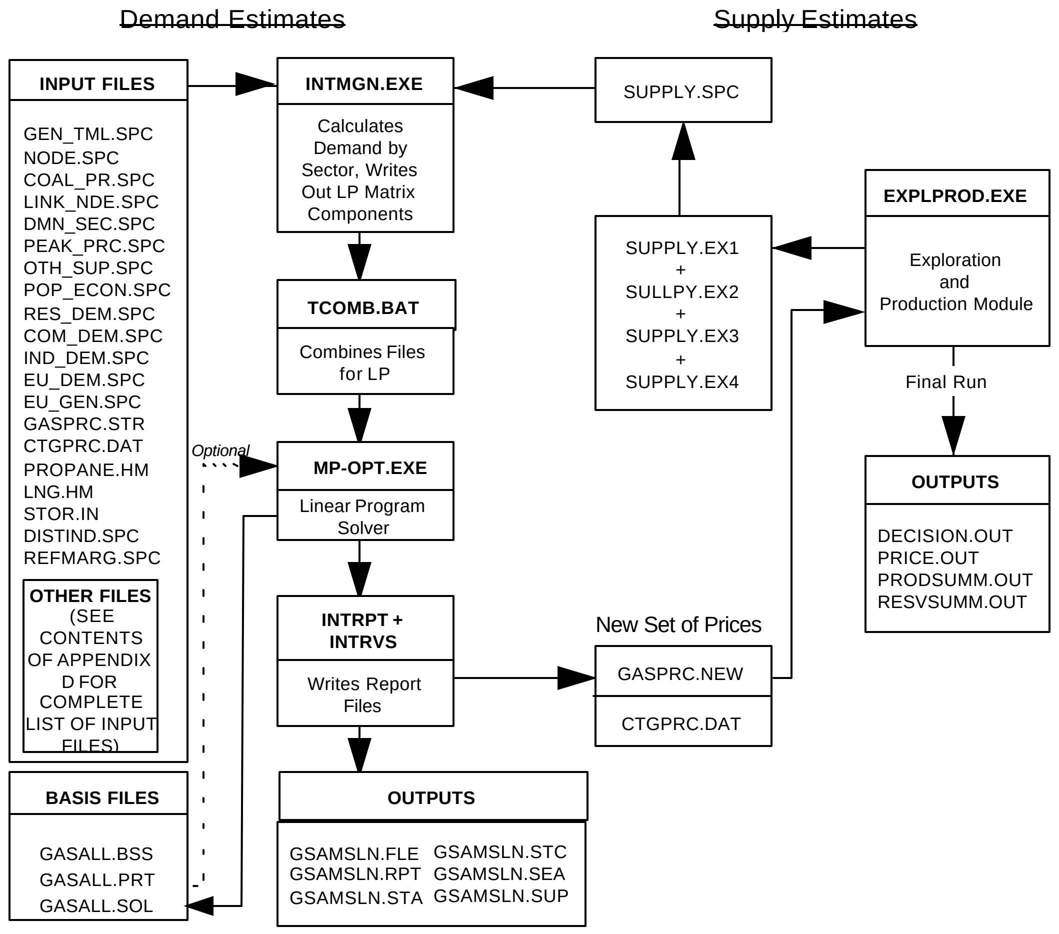
E. Description of Module Components

Figure 7 shows the logic flow for the operation of the Demand and Integrating Modules.

Demand in each region is estimated by sector, season, and year. The model currently uses four seasons, as defined in GEN_TML.SPC. Each sector is evaluated independently using regional considerations as described below.

Demand in the residential and commercial sectors is estimated based on current, base level demand and changes in population and economic factors in the framework of a constant elasticity model. Population and economic growth factors are established for each sector by region for each time specification considered. These calculations are completed in subroutine INTRDD.FOR. Average demand elasticities are estimated using methods developed by EPRI. Seasonal allocation factors from the Uniform Statistical Report are used to divide demand between seasons.

Figure 7
Demand and Integrating Modules



Industrial demand is calculated for each region by econometric models for the Boilers and NUGs subsectors and using aggregate values for the Process Heat and Feedstock subsectors.

Electric utility demand for gas considers the future gas use by the sector as well as fuel competition between gas, coal, and petroleum products in each region. Future demand for gas is calculated based on existing plant demand for gas, coal, and oil, expansion capacity of these plants as electricity demand increases, the relative attractiveness of gas to other fuel alternatives based on seasonal factors, and the type and capacity of new electric utility plants likely to be built in each region. These factors are contained in the file EU_DEM.SPC. EU_GEN.SPC contains information on the costs of running, expanding and building power plants based on their type and capacity. Additional environmental compliance factors are also used to evaluate alternative plant types to be built.

The supply curve data, the demand data, as well as other factors (i.e., storage costs capacity, pipeline costs capacity) are brought together in the integrating linear program. The program INTMGN.EXE combines these data and generates the following .MPS files:

GASCOL.MPS	(related to the non-supply variables in the linear program)
GASCLS.MPS	(related to the supply variables in the linear program)
GASROW.MPS	(related to the constraints of the linear program)
GASRHS.MPS	(related to the non-supply right-hand side data of the linear program)
GASRSS.MPS	(related to the supply right-hand side data of the linear program)
GASBND.MPS	(related to bounds in the non-supply variables)
GASBNS.MPS	(related to bounds in the supply variables)

These .MPS files are then combined (via TCOMB.BAT) into a matrix format called GASALL.MPS which is the input file used by the module's linear program solver.

The linear program solver MP-OPT.EXE reads the matrix file GASSALL.MPS as described above. It considers alternative feasible methods of meeting demand in each region with the supply and transportation options available, and then iterates through solutions to find the optimal solution that balances supply and demand.

Once a valid LP solution is found, the program INTRPT.EXE is called to create various analytical reports. Finally, the program INTRVS.EXE is called to check for convergence of the equilibration process. If it is not achieved, a new GASPRC.NEW file is produced, which is used in the next iteration as the set of market prices in the Exploration and Production Module. If convergence is achieved, a new GASPRC.NEW will not be produced..

F. Frequently Changed Parameters

As well as altering the Exploration and Production Module input parameters in an integrated run, input parameters can also be modified in the Demand and Integrating Module's inputs. A common example is in the LINK_NDE.SPC file in which the pipeline capacities can be modified to gauge the impact of a new pipeline or expansion of an existing pipeline's capacity. This file also contains the start date of potential new projects as well as various costs, see Appendix D, Table D-11 for more detailed information.

VIII. PRODUCTION ACCOUNTING MODULE (5)

A. Summary Description of the Production Accounting Module

The cost processing in the Production Accounting Module is essentially the same as in the Reservoir Performance Module's cost calculation section. The critical difference is that at this stage in the GSAM sequence, market-determined gas prices and production levels have been generated from the Exploration and Production Module. Therefore, there is no need for the Production Accounting Module to calculate results for a range of scenarios. This Module performs the final pro-forma cashflow calculations and aggregates results for states, regions, and the entire United States.

B. Required Files

B.1 Data Input Files

To run the Production Accounting Module, several data files must be read into the system. The following data files are in the \PRODACCT directory:

Name	Description	Location
OUTPUT.OPT	Determines the format of the output files	\PRODACCT
OUTPUT.FED	Similar to OUTPUT.OPT but used for federal runs only	\PRODACCT
OUTPUT.ALL	Similar to OUTPUT.OPT but used for complete (federal and private) runs only	\PRODACCT
COST.DAT	Regional and resource-specific costs and investments for vertical wells, (not same as COST.DAT from Reservoir Performance Module i.e., \RESVPERF\DATA. Here one file contains all regions and horizontal\vertical combination data)	\PRODACCT\DATA
TECH.DAT	Technology parameters for various resources for vertical wells, (same as TECH.DAT from Reservoir Performance Module i.e., \RESVPERF\DATA)	\PRODACCT\DATA
TECH.HOR	Technology parameters for various resources for horizontal wells, (same as TECH.HOR from Reservoir Performance Module i.e., \RESVPERF\DATA)	\PRODACCT\DATA
TAXES.DAT	Severance, income and ad-valorem taxes by State/District, (same as TAXES.DAT from Reservoir Performance Module i.e., \RESVPERF\DATA)	\PRODACCT\DATA
TAX_NAT.DAT	Federal tax specifications, (same as TAX_NAT.DAT from Reservoir Performance Module i.e., \RESVPERF\DATA)	\PRODACCT\DATA
VERHOR.GSM	Output from the selection routine in the Exploration and Production Module contains a list of all undiscovered reservoirs in the run and an indicator whether they were developed with vertical	\PRODACCT\DATA

or horizontal drilling technology

The Production Accounting Module reads certain input files from the Exploration and Production Module. These files do *not* need to be copied to the \PRODACCT directory:

Name	Description	Location
ENV_DAT.SPC	Environmental cost file (P&A module also reads the files specified in ENV_DAT.SPC file)	\EXPLPROD
DECISION.XXX	Decision files created by the Exploration and Production Module	\EXPLPROD
PRICE.XXX	Price files created by the Exploration and Production Module	\EXPLPROD
TAX_CDE.SPC	Tax specifications by year (currently modeled as royalty incentive start year)	\EXPLPROD
PLY_DFN.SPC	Play-specific depth, resource type, royalty rates, development success rates, etc..	\EXPLPROD
GEN_TML.SPC	Time period specifications, discount rate, royalty incentive specifications, model start year, price of crude oil, etc.	\EXPLPROD
ENV_STAT.SPC	Output from environmental file creation program, contains static reservoir information	\EXPLPROD
ENV_PROC.SPC	Output from environmental file creation program, contains gas processing costs by GSAMID	\EXPLPROD
UNDB.BNK	Undiscovered reservoir data bank	\EXPLPROD\NEWS
UNDB.TCP	Undiscovered reservoir data bank	\EXPLPROD\NEWS
DISB.BNK	Discovered reservoir data bank	\EXPLPROD\NEWS
DISB.TCP	Discovered reservoir data bank	\EXPLPROD\NEWS

B.2 Program Files

In addition, the following program files are required:

Name	Description	Location
ACCTPROD.BAT	Batch file which runs entire Production Accounting Module for both federal and federal plus private land runs	\PRODACCT
PRODACCT.EXE	FORTRAN executable which calculates and formats results for Production Accounting Module	\PRODACCT
PSTSRT.EXE	Initial sorting routine	\PRODACCT
POSTARR.EXE	Intermediate sorting routine	\PRODACCT
POSTSORT.EXE	Final sorting routine	\PRODACCT

C. Output Files

The Production Accounting Module can output the following files:

Name	Description	Location
NAT.OUT	Nationally aggregated pro-forma output, non-associated gas only	\PRODACCT
REGION.PRD	Region-level production summaries (if requested in OUTPUT.OPT), non-associated gas only	\PRODACCT
REGION.OUT	Region-level pro-forma output (if requested in OUTPUT.OPT), non-associated gas only	\PRODACCT
REMAIN.OUT	Remaining resource database, non-associated gas only	\PRODACCT
ALL.OUT	Pro-forma output for all reservoirs (if requested in OUTPUT.OPT), non-associated gas only	\PRODACCT
STATE.PRD	State-level production summaries (if requested in OUTPUT.OPT), non-associated gas only	\PRODACCT
STATE.OUT	State-level pro-forma output (if requested in OUTPUT.OPT), non-associated gas only	\PRODACCT

D. Operational Procedures for Running the Production Accounting Module

In the following section we describe operation of the Production Accounting Module both manually and through DOS batch files. Before the Module can be run, the parameters in OUTPUT.OPT must be set. The settings in this file will determine how the output from the Module will be formatted. The choices presented are as follows: (1) write the pro-forma output for every reservoir, (2) aggregate the pro-forma output by State and/or Region, and (3) aggregate the pro-forma output at the national level. In addition, the number of undiscovered reservoirs (number can be calculated in VERHOR.GSM, or can be found in an E&P *LOG file when the E&P Module is run) must be entered manually.

D.1 Manual Operation

NOTE: Although in the Reservoir Performance Module there are different COST files for vertical and horizontal wells in the U.S. and Canada and for vertical wells in Appalachia (as well as a TECH file for that region), there is only one COST file and two TECH files used in the Production Accounting Module (COST.DAT, TECH.HOR, and TECH.DAT). The costs are appropriately taken for U.S. and Canadian reservoirs for vertical and horizontal wells.

a) Make certain all necessary data files from the Reservoir Performance Module are in the PRODACCT\DATA subdirectory (see section B above), verifying that they are the same versions of the files that were used to make the bank files in the E&P Module which produced the decision file to be used in this Production Accounting run. Make sure that COST.DAT file of the Production Accounting Module is consistent with COST.VUS, COST.HUS, COST.VCN, and COST.HCN files

b) Copy \EXPLPROD\PRICE.XXX to \PRODACCT\PRICE.DAT and copy \EXPLPROD\ DECISION.XXX to \PRODACCT\EXNUDS.OUT

c) Run the sorting programs: PSTSRT.EXE, POSTARR.EXE, and POSTSORT.EXE, in that order

d) Run PRODACCT.EXE

D.2 Batch File Operation

Once all of the data files are set appropriately, and a decision is made on the COST files to be employed (see above), the Production Accounting Module is run with ACCTPROD.BAT, which has one argument which defines the extension of the current case (this will be the extension of the DECISION and PRICE files from the Exploration and Production Module). It is run by typing, for example, ACCTPROD OUT, where .OUT is the extension of the decision and price files from the Exploration and Production Module. It is always recommended to run the Production Accounting Module immediately after the Exploration and Production Module for consistency. In order to run ACCTPROD.BAT make sure to have both OUTPUT.FED and OUTPUT.ALL in the \PRODACCT directory, where OUPTUT.FED is for federal land runs and OUTPUT.ALL is for both federal plus private lands runs

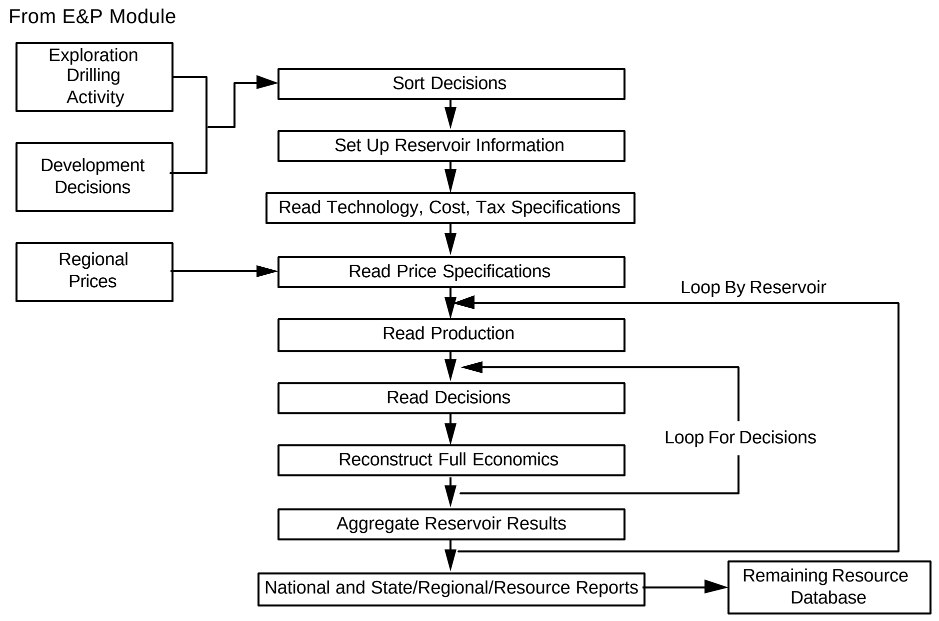
E. Description of Module Components

Upon executing the Module, GSAM calls the sorting routines (PSTSRT.EXE, POSTARR.EXE, POSTSORT.EXE) which sort and match the output data in the DECISION.XXX and PRICE.XXX files which were generated by the Exploration and Production Module. The Module also reads the production files (though the reservoir "banks") generated in the Reservoir Performance Module, instead of running the type curve models again. This provides the information on the production over time needed to complete the final reservoir evaluation.

The performance of the cash flow analysis is virtually the same as that in the Reservoir Performance Module. The purpose of this sequence of procedures is to calculate the total production and total operating costs for each reservoir. Unlike the Reservoir Performance Module, the production and costs need not be developed over a range of price scenarios because at this stage in the GSAM sequence, price has been output by the Exploration and Production Module, either from a stand-alone run, or by generation from the Integrating Module. Figure 8 shows the logic flow and data input for the Production and Accounting Module.

The output from the Module will depend on the settings in the OUTPUT.OPT data file. Pro-forma output for all reservoirs will be sent to ALL.OUT. State/District pro-forma output is placed into REGION.OUT while nationally aggregated output is placed in NAT.OUT. Appendix E presents the formats of these output files. Yearly production values are also reported in the STATE.PRD/REGION.PRD files separately. Finally, the remaining resource database containing all the pertinent information for each reservoir is written into REMAINING.OUT. Note that the Production Accounting Module's outputs contain data for non-associated gas only while the outputs from the Exploration and Production Module contain data for non-associated gas as well as for associated gas.

Figure 8
Production Accounting Module


From E&P Module
Exploration Drilling Activity
Development Decisions
Regional Prices
Sort Decisions
Set Up Reservoir Information
Read Technology, Cost, Tax Specifications
Read Price Specifications
Loop By Reservoir
Read Production
Read Decisions
Loop For Decisions
Reconstruct Full Economics
Aggregate Reservoir Results
National and State/Regional/Resource Reports
Remaining Resource Database

IX. STORAGE RESERVOIR PERFORMANCE MODULE (6)

A. *Summary Description of the Storage Reservoir Performance Module*

The Storage Reservoir Performance Module (SRPM) characterizes storage reservoirs for gas deliverability and injectivity, and associated economics estimations to be used as input data (.SRO) for the Demand and Integrating Module. The modeling concept for generating the extraction and injection response estimates and summary of project economics in the SRPM was adopted from the GSAM's Reservoir Performance (RP) Module with some adjustments and modifications. The RP computer model was modified for gas storage reservoir applications, which include implementation of well injection process and reduction of time step size from one year to one day. The time step adjustment was implemented in the SRPM to provide deliverability and injectivity profiles consistent with the seasonal modeling approach used in the Demand & Integrating Module.

B. *Required Files*

The SRPM utilizes reservoir level properties data (SRPM Database, .STO files) based on American Gas Association (AGA) 1999 release of "Underground Storage of Natural Gas in the U.S. and Canada", and Energy Information Administration (EIA) "U.S. Underground Storage of Natural Gas in 1997: Existing and Proposed" paper. The SRPM database files for U.S. existing and potential storage gas reservoirs are listed below:

B.1 Reservoir Database Files

Name	Description	Location
STODIS.STO	SRPM database for existing storage reservoirs	\SRPM
STOUND.STO	SRPM database for potential storage reservoirs (generated from NRG data)	\SRPM

B.2 Data Input Files

The other data and assumptions required to run the SRPM, including technology specifications, regional production costs, levelized investment costs, and state and federal tax requirements are contained in data files which are also read into the SRPM and listed below:

Name	Description	Location
AFE.DAT	Authorization for expenditure charges data (not currently implemented)	\SRPM\DATA
COST.DAT	Regional and resource-specific costs and investment data	\SRPM\DATA
DWLSPAC.DAT	Well spacing by storage/demand region data to be used as default value if it is not given in the SRPM database	\SRPM\DATA
GEOLOGY.DAT	Specifies reservoir properties by pay-grade distribution (currently only one pay grade is considered in SRPM)	\SRPM\DATA
LEV.DAT	Levelized investment costs and fixed and variable O&M costs by operating company for existing storage reservoirs	\SRPM\DATA
PLAYINFO.DAT	Concentration of gas impurities by play	\SRPM\DATA
ROCKPROP.ADJ	Optional input file containing adjusted reservoir properties data for Technology Run for existing storage reservoirs (copied from STODIS.ADJ)	\SRPM
SROM.TEM	Template file containing description of storage deliverability and injectivity and associated costs information for Demand and Integrating output files (*.SRO)	\SRPM\DATA
TAX_NAT.DAT	Federal tax specifications and other national level tax structures and costs	\SRPM\DATA
TAXES.DAT	Severance, income, and ad valorem taxes by State/District	\SRPM\DATA
TECH.DAT	Technology parameters (only one technology)	\SRPM\DATA
TEMPLATE.DAT	Template file containing description of type curve input parameters, must be specified in REGIONS.DAT to create .TCI files using this template file	\SRPM\DATA

It should be noted that the user may create or modify any of these input files. The SRPM computer model reads in numbers in the data files as free-format so the position, decimal places, etc. of the data does not have to be specified at a particular location in the file. Note also that data files such as COST.DAT and TECH.DAT have headers throughout the data, and whether these files are modified or recreated, these headers must remain in the same format.

B.3 Run Specification Files

The following files contain instructions on the type and configuration of the SRPM run being conducted. The SRPM computer model utilizes the information contained in these files to set up the formats, identify inputs, and set up key parameters. The files include:

Name	Description	Location
REGIONS.DAT	Specifies the file names of storage databases (*.STO) to be run through the SRPM and YES/NO switches as indicators for creating specific report files	\SRPM\DATA
SRPMSPEC.DAT	SRPM run specifications such as name of directory for output files, type of run for existing storage reservoirs, number of years and maximum working gas capacity (fraction of OGIP) for potential storage reservoirs	\SRPM

B.4 Program Files

Name	Description	Location
SRPM.EXE	FORTRAN executable file which will run the Storage Reservoir Performance Module	\SRPM

Appendix F contains printouts of the input data and run specification files currently being used in the GSAM Storage Reservoir Performance Module. Again, these files can be modified to conduct individual analyses by changing the storage reservoir performance parameters and/or changes in operating costs.

C. Output Files

The Storage Reservoir Performance Module will create the following main output files. To ensure that these files are not overwritten in a subsequent run of the Module, be sure to specify different output file directory name in the SRPMSPEC.DAT file:

Name	Description	Location
STODIS.ADJ	Adjusted reservoir properties data generated from Non-Technology Run (i.e. Base run) for existing storage reservoirs; must be copied to \SRPM\ROCKPROP.ADJ if Technology Run on existing storage reservoirs needs to be performed	\SRPM\[OUTPUT DIRECTORY]
STODIS.SRO	Existing storage reservoir performance output to be used in Demand and Integrating Module	\SRPM\[OUTPUT DIRECTORY]
STOUND.SRO	Potential storage reservoir performance output to be used in Demand and Integrating Module	\SRPM\[OUTPUT DIRECTORY]
.ERR	Output file reporting error/action messages on a reservoir level	\SRPM\[OUTPUT DIRECTORY]

Additional reservoir-level summary files are created if requested in REGIONS.DAT. These include:

Name	Description	Location
.PRD	Optional output file containing summary or flow rates and pressures on a reservoir level; it is created only if its flag in REGIONS.DAT is set to "YES"	\SRPM\[OUTPUT DIRECTORY]
.PRO	Optional output file containing detailed cash flow pro-forma on a reservoir level; it is created only if its flag in REGIONS.DAT is set to "YES"	\SRPM\[OUTPUT DIRECTORY]
.TCI	Optional output file containing type curve input parameters on a reservoir level; it is created only if its flag in REGIONS.DAT is set to "YES"	\SRPM\[OUTPUT DIRECTORY]
.TCO	Optional output file containing type curve output on a reservoir level; it is created only if its flag in REGIONS.DAT is set to "YES"	\SRPM\[OUTPUT DIRECTORY]

D. Operational Procedures for Running the Storage Reservoir Performance Module

The Storage Reservoir Performance Module can be run using one of the following options:

D.1 Non-Technology Run (Base Run) for Existing Storage Reservoirs:

The following procedures are needed to obtain a consistent set of Working Gas and Base Gas values at the specified set of rock and fluid properties, well properties, and reservoir data.

In this mode, the SRPM performs two types of reservoir property adjustments. The first adjustment is performed prior to performing the reservoir simulation (type curve). At this stage, the SRPM adjusts porosity, saturation, pay thickness, well drainage area, and well spacing of the existing reservoirs (STODIS.STO) to match the calculated original gas in place (OGIP) with the ultimate storage capacity reported by the American Gas Association (AGA). The second step is to adjust permeability and skin factor to match the maximum working gas capacity reported by the AGA which involves trial and error process requiring one simulation run (type curve) for each iteration. The final adjusted reservoir properties are then stored in output file \SRPM\[OUTPUT DIRECTORY]\STODIS.ADJ and the storage reservoir performance results are stored in output file \SRPM\[OUTPUT DIRECTORY]\STODIS.SRO. The running procedure for the Non-Technology Run option is as follows:

- Make sure the existing storage reservoir database (i.e. STODIS.STO) exists in the main SRPM directory
- Specify STODIS as the prefix of the database file in input file SRPM\DATA\ REGIONS.DAT
- Set flag of run type in input file \SRPM\SRPMSPEC.DAT (line #4) to 0
- Run SRPM.EXE

D.2 Technology Run for Existing Storage Reservoirs:

This run is conducted for the purpose of evaluating the effect of implementing different well technology specifications (such as fracturing, horizontal wells, acidizing, etc. cases) as specified in input file \SRPM\DATA\TECH.DAT. In this mode, the SRPM utilizes the adjusted reservoir properties (existing reservoirs) that were previously generated by the SRPM run with "Non-Technology" option and performs single simulation run (type curve) for each reservoir. The storage reservoir performance results are stored in output file \SRPM\[OUTPUT DIRECTORY]\STODIS.SRO. The running procedure for the Technology Run option is as follows:

- Make sure the existing storage reservoir database (i.e. STODIS.STO) exists in the main SRPM directory
- Specify STODIS as the prefix of the database file in input file SRPM\DATA\ REGIONS.DAT
- Copy \SRPM\[OUTPUT DIRECTORY]\STODIS.ADJ to \SRPM\ ROCKPROP.ADJ
- Set flag of run type in input file \SRPM\SRPMSPEC.DAT (line #4) to 1
- Run SRPM.EXE

D.3 Run for Potential Storage Reservoirs:

In this mode, the SRPM adjusts well spacing of the potential reservoirs (STOUND.STO) to match user specified maximum working gas capacity (a fraction of OGIP) given in input file \SRPM\SRPMSPEC.DAT. This is an iterative process requiring one simulation run (type curve) for each iteration. The storage reservoir performance results are stored in output file

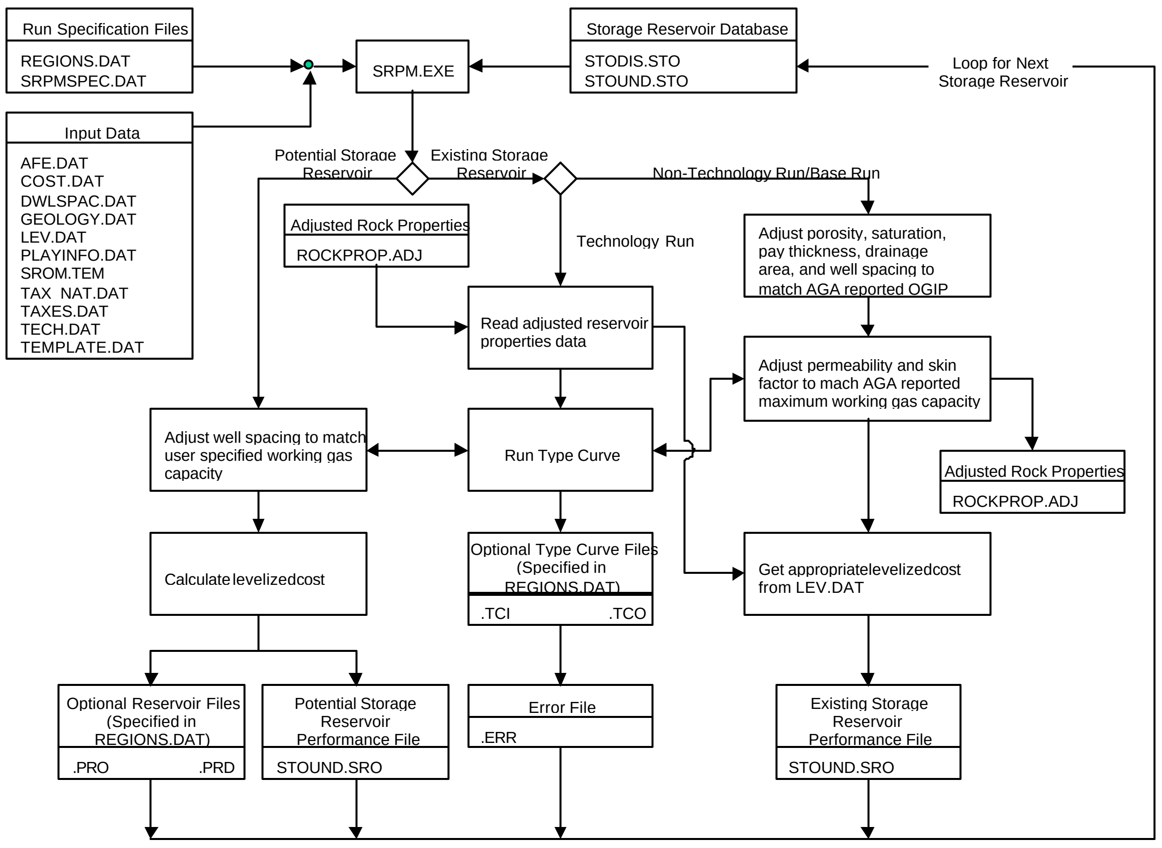
\SRPM\[OUTPUT DIRECTORY]\STOUND.SRO. The running procedure for Potential Storage Reservoir Run is as follows:

- Make sure the potential storage reservoir database (i.e. STOUND.STO) exists in the main SRPM directory
- Specify STOUND as the prefix of the database file in input file SRPM\DATA\REGIONS.DAT
- Specify the expected maximum working gas capacity in the unit of fraction of OGIP in line #8 of input file \SRPM\SRPMSPEC.DAT
- Set number of years (line #6 of input file \SRPM\SRPMSPEC.DAT) for the operation of the potential storage reservoirs to be used in the calculations of levelized investment costs and fixed and variable O&M costs.
- Run SRPM.EXE

E. Description of Module Components and Programs

The Storage Reservoir Performance Module calculates the summary economics and extraction/injection response estimates for each individual storage reservoir on a sequential basis. The details of the program are described below in Figure 9.

Figure 9
Flow Chart for Storage Reservoir Performance Module



Run Specification Files
REGIONS.DAT
SRPMSPEC.DAT
Input Data
AFE.DAT
COST.DAT
DWLSPAC.DAT
GEOLOGY.DAT
LEV.DAT
PLAYINFO.DAT
SROM.TEM
TAX NAT.DAT
TAXES.DAT
TECH.DAT
TEMPLATE.DAT
SRPM.EXE
Storage Reservoir Database
STODIS.STO
STOUND.STO
Loop for Next Storage Reservoir
Potential Storage Reservoir
Existing Storage Reservoir
Non-Technology Run/Base Run
Technology Run
Adjusted Rock Properties
ROCKPROP.ADJ
Read adjusted reservoir properties data
Adjust porosity, saturation, pay thickness, drainage area, and well spacing to match AGA reported OGIP
Adjust permeability and skin factor to mach AGA reported maximum working gas capacity
Adjust well spacing to match user specified working gas capacity
Run Type Curve
Adjusted Rock Properties
ROCKPROP.ADJ
Calculate levelizedcost
Optional Type Curve Files (Specified in REGIONS.DAT)
.TCI
.TCO
Get appropriatelevelizedcost from LEV.DAT
Optional Reservoir Files (Specified in REGIONS.DAT)
.PRO
.PRD
Potential Storage Reservoir Performance File
STOUND.SRO
Error File
.ERR
Existing Storage Reservoir Performance File
STOUND.SRO

E.1 Read in Data

Once the Storage Reservoir Performance Module (SRPM) is invoked, SRPM selects the first reservoir and extracts data from the SRPM database (.STO files) specified in REGIONS.DAT and determines whether the reservoir in question is an existing storage reservoir or a potential storage reservoir. For existing storage reservoir with Non-Technology Run option (specified in \SRPM\SRPMSPEC.DAT), adjustments to some reservoir properties such as porosity, saturation, pay thickness, drainage area, and well spacing are performed to match the calculated original gas in place (OGIP) with the ultimate storage capacity value reported by the American Gas Association (AGA).

E.2 Type Curve Module

The next step after acquiring the storage reservoir properties from the database is to perform reservoir simulation (type curve) to each storage reservoir. Regardless of the type of run specified in \SRPM\SRPMSPEC.DAT, each storage reservoir will undergo calculations of deliverability, injectivity, and project economic based on a single set of technology, price, and costs. The results of these calculations are stored before the next storage reservoir is analyzed. For existing storage reservoir with Technology Run option, the type curve calculation is straight forward as no further adjustment to the reservoir properties is required. For the two other options, (existing storage reservoir with Non-Technology Run option and potential storage reservoir), the type curve calculation involves trial and error process because the calculation requires further adjustments to some of the reservoir properties that affects the type curve responses. The existing storage reservoir with Non-Technology Run option requires adjustments to permeability and skin factor to match the AGA reported value of maximum working gas capacity. The SRPM run for potential storage reservoirs requires adjustment to well spacing to match the user specified working gas capacity (given in \SRPM\SRPMSPEC.DAT).

The SRPM type curve module can simulate any of the six different storage reservoir systems to predict deliverability and injectivity of the reservoir. The six storage reservoir systems are: (1) radial flow in conventional gas reservoirs, (2) linear flow in conventional gas reservoirs (i.e. with hydraulic fractures), (3) radial flow in naturally fractured gas reservoirs, (4) linear flow in naturally fractured gas reservoirs (i.e. with hydraulic fractures), (5) radial flow in water drive gas reservoirs, and (6) salt dome reservoir (currently modeled as a conventional reservoir with very high porosity and permeability). The type curve module currently solves only one development case (primary well case) and one pay grade (pay grade #2). The module considers two production/injection modes:

1. Storage mode: specifying fixed production and injection cycles within one-year (365 days) period with one-day time step size. The module only solves for production cycle (currently 120 days production period) and assumes that the injection cycle can bring the system pressure to the initial reservoir pressure by injecting "working gas" volume at the end of the year.

2. Full production mode: no injection is performed in this mode. The SRPM is run using a daily basis time step sizes for a specified number of days (currently 120 days) with the purpose to determine base gas, working gas, etc. for potential storage facility.

The gathering pressure (wellhead pressure) is assumed to be the same for all wells in the field. The storage reservoir is allowed to produce against a minimum allowable wellhead pressure constraint as long as the total gas production rate does not exceed the maximum allowable total gas rate. Otherwise, the maximum allowable total gas rate constraint is utilized.

E.3 Cash Flow Analysis

After the type curve process has been completed, the Storage Reservoir Performance Module performs cash flow analyses. Costs are set based on information in COST.DAT, including drilling and completion, overhead, compression, and fixed and variable O&M. Specific tax rates used in the economic evaluations are set in TAXES.DAT and TAX_NAT.DAT for state and federal parameters. The results of production and operating costs for each storage reservoir are then stored in .PRD files and the detailed financial results for each storage reservoir are reported in output file .PRO.

F. Frequently Changed Parameters

The nature of the Storage Reservoir Performance Module is such that not more than a few input parameters will most likely be altered. Two files, TECH.DAT and COST.DAT, allow for manipulation (remember, these files correspond to specific regions). In the COST.DAT file, the drilling costs, operating costs, discount rate, and various cost factors can be changed. In TECH.DAT, the skin factor as well as many other technologies parameters can be set. To investigate the sensitivity, certain parameters from these files can be changed such as the entries in COST.DAT and TECH.DAT files and executing SRPM.EXE executable. As always, the user should use different output directory name (specified in \SRPM\SRPMSPEC.DAT) for each run.

APPENDIX A
RESOURCE MODULE FILES

CONTENTS

Table A-1

File: GSAM##.GSM (Location: \GSAM\RESOURCE\USDISC)
This is the discovered U.S. database based on the 1997 NRG Associates database. The formatted NRG files are called GSAM##.GSM, where "##" is a 2-digit number ranging from 01 to 12 and from 15 to 19.

Data Element	SAS Position	Decimal	FORTRAN Format	Description
Record 1				
GSAMID	1-12		12A,1X,	Unique GSAM Identification Number
RESCOD	14-20		I8,1X,	Reservoir Code
FLDNAM	22-51		30A,1X,	Field Name
EIACOD	53-62		10A,1X,	EIA Field Code
RSVNAM	64-93		30A,1X,	Reservoir Formation Name
STATE	95-98		I4,1X,	State Code
COUNTY	100-104		I5,1X,	County Code
PLYCOD	106-110		I5,1X,	Play Code
BSNCOD	112-114		I3,1X,	Basin Code
FSCLAS	116-118		I3	Field Size Class
(end of line)				
Record 2				
TWNSHP	1-8		A8,1X,	Township Name
RANGE	10-17		A8,1X,	Range Name
ONOFFS	19-20		I2,1X,	Onshore/offshore (1=ONSHORE, 2=STATE OFFSHORE, 3=FEDERAL OFFSHORE
GSAMSR	22-25		I4,1X,	GSAM Supply Region
STATIN	27-28		I2,1X,	Initial Development Status
LAT	30-36	.3	F7.3,1X,	Latitude of Reservoir Centroid
LON	38-44	.3	F7.3,1X,	Longitude of Reservoir Centroid
DEPCLS	46-49		I4,1X,	Depositional Class
RESTYPE	51-54		I4,1X,	Lithology Type
TRAPTY	56-57		I2,1X,	Trap Type
DRIVE	59-60		I2,1X,	Dominant Drive Type
DEPTH	62-67		F7.0,	Depth (feet)
DEPTSS	69-74		F7.0,	Depth Subsea (feet)
H2Odep	76-81		F7.0	Water Depth (feet)
(end of line)				
Record 3				
GRSPAY	1-5		Format free,	Gross Pay Thickness (feet)
NETPAY	7-11 Formation (feet)		Format free,	Total Net Pay in Designated
PAYDSP	13-15	.1	Format free,	Pay Dispersion Function
WELDRN	17-21	.3	Format free,	Well Drainage Area (acres)
PERHOR	23-32	.3	Format free,	Effective Horizontal Permeability (md)
PERVRT	34-43	.3	Format free,	Effective Vertical Permeability (md)
PERMTX	45-54	.3	Format free,	Matrix Permeability(md)

Table A-1 (continued)

File: GSAM##.GSM

Data Element	SAS Position	Decimal	FORTRAN Format	Description
PORTOT	56-60	.3	Format free,	Total Effective Initial Porosity
PORMTX	62-66	.3	Format free,	Matrix Porosity
PORCUR	68-72	.3	Format free,	Current Total Effective Porosity
WATSAT	74-78	.2	Format free,	Initial Water Saturation
GASSAT	80-84	.2	Format free	Initial Gas Saturation
(end of line)				
Record 4				
PRESIN	1-6		Format free,	Initial Reservoir Shut-in Pressure(PSIA)
GASGRV	8-12	.3	Format free,	Specific Gravity of Dry Gas
BHTEMP	14-18		Format free,	Bottomhole Temperature(degrees F)
HEATVL	20-24		Format free,	Heating Value (BTU/CF)
CO2	26-31	.3	Format free,	Carbon Dioxide Contamination Fraction
N2	33-38	.3	Format free,	Nitrogen Contamination Fraction
H2S	40-45	.3	Format free,	Hydrogen Sulfide Contamination Fraction
WELRAD	47-51	.3	Format free,	Wellbore Radius (feet)
COMPFR	53-61	.7	Format free,	Formation Compressibility
SOLGAS	63-68		Format free,	Gas Solubility in Brine
CHLCON	70-76		Format free,	Cl Concentration of Produced Water
CMPWAT	78-86	.7	Format free,	Water Compressibility
INPORF	88-91	.2	Format free,	Interporosity Flow Factor
LANGVL	93-98	.1	Format free,	Langmuir Volume
LANGPR	100-105	.1	Format free,	Langmuir Pressure
PRSDSP	107-111	.1	Format free,	Desorption Pressure
GASCON	113-118	.2	Format free,	Initial Gas Concentration
UNCTYPE	120		Format free,	Coal/Shale type (Wet/Dry)
UNCLOC	122		Format free	Coal/Shale Location (East/West)
(end of line)				
Record 5				
SRPTIM	1-6	.1	Format free,	Pseudo Steady State Desorption Time (days)
AQURAD	8-14		Format free,	Aquifer Radius (feet)
AQUPRM	16-25	.3	Format free,	Aquifer Permeability (md)
FRACSP	27-33	.3	Format free,	Natural Fracture Spacing (feet)
FRACWI	35-39	.3	Format free,	Natural Fracture Width (feet)
FRACFL	41-50	.5	Format free,	Fracture Flow Parameter
FRACXF	52-57		Format free,	Fracture Half Length or Length of Contact (feet)
FRACCN	59-66		Format free,	Induced Fracture Conductivity (md-feet)
FRACSK	68-73	.1	Format free,	Induced Fracture Skin Factor
FRACPO	75-84	.3	Format free,	Induced Fracture Porosity

Table A-1 (continued)

File: GSAM##.GSM

Data Element	SAS Position	Decimal	FORTRAN Format	Description

```
    WATSAF    86-89       .2     Format free,   Fracture Water Saturation
    DISCYR    91-95              Format free,   Date of Reservoir Discovery
    DISFLD    97-101             Format free,   Date of Field Discovery
    DISMTH    103-104            Format free,   Reservoir Discovery Method
    SGDVYR    106-110            Format free,   Year Significant Development
                                                Drilling Starts
    DOMOPR    112-117            Format free,   Dominant Operator as of 1997
    WLSPAC    119-123            Format free    Well Spacing (Current Field Rule)
    (end of line)

Record 6
    ACPROD    1-8                Format free,   Estimated Total Production
                                                Area (acres)
    ACPROV    10-17              Format free,   Maximum Proved Area (acres)
    ACPRVD    19-23              Format free,   Date of Maximum Proved
                                                Area Estimate
    ACDV97    25-32              Format free,   Reservoir Developed Area EOY 1997
    OGIP      34-42       .3     Format free,   Reservoir Volumetric Original
                                                Gas in Place
    CGPR97    44-52       .3     Format free,   Cumulative Gas Production
                                                to 1997 (Bcf)
    GRSV97    54-62       .3     Format free,   Proved Gas Reserves End of
                                                1997 (Bcf)
    RSVCLS    64-66              Format free,   AAPG Reservoir Size Class
    GWR97     68-73       .1     Format free,   1997 Gas-Water Ratio
    PRSCUR    75-80              Format free,   Current Bottomhole Shut-in
                                                Pressure
    WATSAC    82-86       .2     Format free,   Current Water Saturation
    WATSAB    88-92       .2     Format free,   Abandonment Water Saturation
    PRSFLW    94-99              Format free,   Current Bottomhole Flowing
                                                Pressure
    PRORAT    101-109     .2     Format free,   Proration - Rule for
                                                Wells/Reservoir
    NGLFACT   111-121     .3     Format free,   Barrels NGL/MMcf dry gas
    (end of line)

Record 7
    GASPRD82  1-8         .3     Format free,   1982 Gas Production (Bcf)
    GASPRD83  9-16        .3     Format free,   1983 Gas Production (Bcf)
    GASPRD84  17-24       .3     Format free,   1984 Gas Production (Bcf)
    GASPRD85  25-32       .3     Format free,   1985 Gas Production (Bcf)
    GASPRD86  33-40       .3     Format free,   1986 Gas Production (Bcf)
    GASPRD87  41-48       .3     Format free,   1987 Gas Production (Bcf)
    GASPRD88  49-56       .3     Format free,   1988 Gas Production (Bcf)
    GASPRD89  57-64       .3     Format free,   1989 Gas Production (Bcf)
    GASPRD90  65-72       .3     Format free,   1990 Gas Production (Bcf)
    GASPRD91  73-80       .3     Format free,   1991 Gas Production (Bcf)
    GASPRD92  81-88       .3     Format free,   1992 Gas Production (Bcf)
    GASPRD93  89-96       .3     Format free,   1993 Gas Production (Bcf)
    GASPRD94  97-104      .3     Format free,   1994 Gas Production (Bcf)
    GASPRD95  105-112     .3     Format free,   1995 Gas Production (Bcf)
```

Table A-1 (continued)

File: GSAM##.GSM

```
                 SAS              FORTRAN
Data Element     Position    Decimal      Format         Description

    GASPRD96  113-120     .3     Format free,   1996 Gas Production (Bcf)
    GASPRD97  121-128     .3     Format free,   1997 Gas Production (Bcf)
    (end of line)
```

```
Record 8
    OILPRD82  1-8               Format free,  1982 Oil Production (Mbbls)
    OILPRD83  9-16              Format free,  1983 Oil Production (Mbbls)
    OILPRD84  17-24             Format free,  1984 Oil Production (Mbbls)
    OILPRD85  25-32             Format free,  1985 Oil Production (Mbbls)
    OILPRD86  33-40             Format free,  1986 Oil Production (Mbbls)
    OILPRD87  41-48             Format free,  1987 Oil Production (Mbbls)
    OILPRD88  49-56             Format free,  1988 Oil Production (Mbbls)
    OILPRD89  57-64             Format free,  1989 Oil Production (Mbbls)
    OILPRD90  65-72             Format free,  1990 Oil Production (Mbbls)
    OILPRD91  73-80             Format free,  1991 Oil Production (Mbbls)
    OILPRD92  81-88             Format free,  1992 Oil Production (Mbbls)
    OILPRD93  89-96             Format free,  1993 Oil Production (Mbbls)
    OILPRD94  97-104            Format free,  1994 Oil Production (Mbbls)
    OILPRD95  105-112           Format free,  1995 Oil Production (Mbbls)
    OILPRD96  113-120           Format free,  1996 Oil Production (Mbbls)
    OILPRD97  121-128           Format free,  1997 Oil Production (Mbbls)
    (end of line)

Record 9
    NGLPRD82  1-8               Format free,  1982 NGL Production (Mbbls)
    NGLPRD83  9-16              Format free,  1983 NGL Production (Mbbls)
    NGLPRD84  17-24             Format free,  1984 NGL Production (Mbbls)
    NGLPRD85  25-32             Format free,  1985 NGL Production (Mbbls)
    NGLPRD86  33-40             Format free,  1986 NGL Production (Mbbls)
    NGLPRD87  41-48             Format free,  1987 NGL Production (Mbbls)
    NGLPRD88  49-56             Format free,  1988 NGL Production (Mbbls)
    NGLPRD89  57-64             Format free,  1989 NGL Production (Mbbls)
    NGLPRD90  65-72             Format free,  1990 NGL Production (Mbbls)
    NGLPRD91  73-80             Format free,  1991 NGL Production (Mbbls)
    NGLPRD92  81-88             Format free,  1992 NGL Production (Mbbls)
    NGLPRD93  89-96             Format free,  1993 NGL Production (Mbbls)
    NGLPRD94  97-104            Format free,  1994 NGL Production (Mbbls)
    NGLPRD95  105-112           Format free,  1995 NGL Production (Mbbls)
    NGLPRD96  113-120           Format free,  1996 NGL Production (Mbbls)
    NGLPRD97  121-128           Format free,  1997 NGL Production (Mbbls)
(end of line)
```

Table A-1 (continued)

File: GSAM##.GSM

Data Element	SAS Position	Decimal	FORTRAN Format	Description
Record 10				
TOTWEL82	1-6		Format free,	1982 Total Wells
TOTWEL83	7-12		Format free,	1983 Total Wells
TOTWEL84	13-18		Format free,	1984 Total Wells
TOTWEL85	19-24		Format free,	1985 Total Wells
TOTWEL86	25-30		Format free,	1986 Total Wells
TOTWEL87	31-36		Format free,	1987 Total Wells
TOTWEL88	37-42		Format free,	1988 Total Wells
TOTWEL89	43-48		Format free,	1989 Total Wells
TOTWEL90	49-54		Format free,	1990 Total Wells
TOTWEL91	55-60		Format free,	1991 Total Wells
TOTWEL92	61-66		Format free,	1992 Total Wells
TOTWEL93	67-72		Format free,	1993 Total Wells
TOTWEL94	73-78		Format free,	1994 Total Wells
TOTWEL95	79-84		Format free,	1995 Total Wells
TOTWEL96	85-90		Format free,	1996 Total Wells
TOTWEL97	91-96		Format free,	1997 Total Wells
(end of line)				
Record 11				
PRDWEL82	1-6		Format free,	1982 Producing Wells
PRDWEL83	7-12		Format free,	1983 Producing Wells
PRDWEL84	13-18		Format free,	1984 Producing Wells
PRDWEL85	19-24		Format free,	1985 Producing Wells
PRDWEL86	25-30		Format free,	1986 Producing Wells
PRDWEL87	31-36		Format free,	1987 Producing Wells
PRDWEL88	37-42		Format free,	1988 Producing Wells
PRDWEL89	43-48		Format free,	1989 Producing Wells
PRDWEL90	49-54		Format free,	1990 Producing Wells
PRDWEL91	55-60		Format free,	1991 Producing Wells
PRDWEL92	61-66		Format free,	1992 Producing Wells
PRDWEL93	67-72		Format free,	1993 Producing Wells
PRDWEL94	73-78		Format free,	1994 Producing Wells
PRDWEL95	79-84		Format free,	1995 Producing Wells
PRDWEL96	85-90		Format free,	1996 Producing Wells
PRDWEL97	91-96		Format free,	1997 Producing Wells
(end of line)				
Record 12				
SHUTWL82	1-6		Format free,	1982 Shut-in Wells
SHUTWL83	7-12		Format free,	1983 Shut-in Wells
SHUTWL84	13-18		Format free,	1984 Shut-in Wells
SHUTWL85	19-24		Format free,	1985 Shut-in Wells
SHUTWL86	25-30		Format free,	1986 Shut-in Wells
SHUTWL87	31-36		Format free,	1987 Shut-in Wells
SHUTWL88	37-42		Format free,	1988 Shut-in Wells
SHUTWL89	43-48		Format free,	1989 Shut-in Wells

Table A-1 (continued)

File: GSAM##.GSM

	SAS	FORTRAN		
Data Element	Position	Decimal	Format	Description
SHUTWL90	49-54		Format free,	1990 Shut-in Wells
SHUTWL91	55-60		Format free,	1991 Shut-in Wells
SHUTWL92	61-66		Format free,	1992 Shut-in Wells
SHUTWL93	67-72		Format free,	1993 Shut-in Wells
SHUTWL94	73-78		Format free,	1994 Shut-in Wells
SHUTWL95	79-84		Format free,	1995 Shut-in Wells
SHUTWL96	85-90		Format free,	1996 Shut-in Wells
SHUTWL97	91-96		Format free,	1997 Shut-in Wells
(end of line)				
Record 13				
TWLSPAC	1-5		Format free,	Target Well Spacing
BPSLOP	7-11	.3	Format free,	Backpressure Exponent
PRSSYS	13-18		Format free,	Operating System Back Pressure (PSIA)
PZSLOP	20-28	.2	Format free,	Slope of Cumulative Production vs. p/z
FLDTYPE	30-31		Format free,	Type of Field
MODULE1	33		Format free,	Type Curve Module to use
FRAC_FED	35		Format free,	Flag for Private (0) or Federal (1) land
(end of line)				

Sample Reservoir Data for a U.S. Reservoir GSAMID: 07314401P001

```
07314401P001       0 PAYTON                            545289     MSSP POOL                     4280 42475 4401  430   5
NONE      NONE      1 07    3 31.312  102.880    1   35  6  0   6412   3973      0
   38    19 2.0 160.0     22.989      6.897     22.989 0.058 0.058 0.058  0.23  0.77
  2926 0.652   136   960  0.000  0.000  0.000 0.354 0.0000030      0   30000 0.0000030 0.00    0.0    0.0   0.0   0.00 0 0
   0.0      0     22.989   0.000 0.000    0.00000      0         0    0.0      0.000 0.00 1956  1937   8 1956  2250     160
    1929     7000  1997     7000    14.956     4.134     0.216 1      0.0      0  0.23  0.30      0      0.25      24.138
   0.003   0.043   0.013   0.005   0.012   0.013   0.018   0.020   0.035   0.015   0.006   0.010   0.020   0.013   0.011
0.043
       0       0       0       0       0       0       0       0       0       0       0       0       0       0       0
0
       0       1       0       0       0       1       0       0       1       1       0       0       0       0       0
1
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   80 0.850    800      0.00 1  1 0
```

Table A-2

File: CANADA.GSM, CANDU.GSM (Location: \GSAM\RESOURCE\CANDISC)
This is the discovered Canadian database. The files are CANADA.GSM and CANDU.GSM. CANADA.GSM contains data for the discovered producing reservoirs. CANDU.GSM contains data for the discovered undeveloped reservoirs. These are derived from the 1994 version of the NRG Associates database. The GSAMID-s of all Canadian reservoirs are manually updated to 12-digit (by inserting “p” at the end of the play ID) for consistency purposes. Note that the Canadian Resource Module creates 11-digit GSAMID-s

Data Element	SAS Position	Decimal	FORTRAN Format	Description
Record 1				
GSAMID	1-12		12A,1X,	Unique GSAM Identification Number
RESCOD	13-20		I8,1X,	Reservoir Code
FLDNAM	22-51		30A,1X,	Field Name
EIACOD	53-62		10A,1X,	EIA Field Code
RSVNAM	64-93		30A,1X,	Reservoir Formation Name
STATE	95-98		I4,1X,	State Code
COUNTY	100-104		I5,1X,	County Code
PLYCOD	106-110		I5,1X,	Play Code
BSNCOD	112-114		I3,1X,	Basin Code
FSCLAS	116-118		I3	Field Size Class
(end of line)				
Record 2				
TWNSHP	1-8		A8,1X,	Township Name
RANGE	10-17		A8,1X,	Range Name
ONOFFS	19-20		I2,1X,	Onshore/offshore (1=ONSHORE, 2=OFFSHORE)
GSAMSR	22-25		I4,1X,	GSAM Supply Region
STATIN	27-28		I2,1X,	Initial Development Status
LAT	30-36	.3	F7.3,1X,	Latitude of Reservoir Centroid
LONG	38-44	.3	F7.3,1X,	Longitude of Reservoir Centroid
DEPCLS	46-49		I4,1X,	Depositional Class
RESTYPE	51-54		I4,1X,	Lithology Type
TRAPTY	56-57		I2,1X,	Trap Type
DRIVE	59-60		I2,1X,	Dominant Drive Type
DEPTH	62-67		F7.0,	Depth (feet)
DEPTSS	69-74		F7.0,	Depth Subsea (feet)
H2ODEP			F7.0,	Water Depth (feet)
(end of line)				
Record 3				
GRSPAY	1-5		Format free,	Gross Pay Thickness (feet)
NETPAY	7-11		Format free,	Total Net Pay in Designated Formation (feet)
PAYDSP	13-15	.1	Format free,	Pay Dispersion Function
WELDRN	17-21	.3	Format free,	Well Drainage Area (acres)
PERHOR	23-32	.3	Format free,	Effective Horizontal Permeability (md)
PERVRT	34-43	.3	Format free,	Effective Vertical Permeability (md)
PERMTX	45-54	.3	Format free,	Matrix Permeability (md)
PORTOT	56-60	.3	Format free,	Total Effective Initial Porosity
PORMTX	62-66	.3	Format free,	Matrix Porosity
PORCUR	68-72	.3	Format free,	Current Total Effective Porosity

Table A-2 (continued)

File: CANADA.GSM, CANDU.GSM

Data Element	SAS Position	Decimal	FORTRAN Format	Description
WATSAT	74-78	.2	Format free,	Initial Water Saturation
GASSAT	80-84	.2	Format free	Initial Gas Saturation
(end of line)				
Record 4				
PRESIN	1-6		Format free,	Initial Reservoir Shut-in Pressure (PSIA)
GASGRV	8-12	.3	Format free,	Specific Gravity of Dry Gas
BHTEMP	14-18		Format free,	Bottomhole Temperature (degrees F)
HEATVL	20-24		Format free,	Heating Value (BTU/CF)
CO2	26-31	.3	Format free,	Carbon Dioxide Contamination
N2	33-38	.3	Format free,	Nitrogen Contamination
H2S	40-45	.3	Format free,	Hydrogen Sulfide Contamination
WELRAD	47-51	.3	Format free,	Wellbore Radius (feet)
COMPFR	53-61	.7	Format free,	Formation Compressibility
SOLGAS	63-68		Format free,	Gas Solubility in Brine
CHLCON	70-76		Format free,	Cl Concentration of Produced Water
CMPWAT	78-86	.7	Format free,	Water Compressibility
INPORF	88-91	.2	Format free,	Interporosity Flow Factor
LANGVL	93-98	.1	Format free,	Langmuir Volume
LANGPR	100-105	.1	Format free,	Langmuir Pressure
PRSDSP	107-111	.1	Format free,	Desorption Pressure
GASCON	113-118	.2	Format free,	Initial Gas Concentration
UNCTYPE	120		Format free,	Coal/Shale type (Wet Dry)
UNCLOC	122		Format free	Coal/Shale Location (East West)
(end of line)				
Record 5				
SRPTIM	1-6	.1	Format free,	Pseudo Steady State Desorption Time (days)
AQURAD	8-14		Format free,	Aquifer Radius (feet)
AQUPRM	16-25	.3	Format free,	Aquifer Permeability (md)
FRACSP	27-33	.3	Format free,	Natural Fracture Spacing (feet)
FRACWI	35-39	.3	Format free,	Natural Fracture Width (feet)
FRACFL	41-50	.5	Format free,	Fracture Flow Parameter
FRACXF	52-57		Format free,	Fracture Half Length or Length of Contact (ft)
FRACCN	59-66		Format free,	Induced Fracture Conductivity (md-ft)
FRACSK	68-73	.1	Format free,	Induced Fracture Skin Factor
FRACPO	75-84	.3	Format free,	Induced Fracture Porosity
WATSAF	86-89	.2	Format free,	Fracture Water Saturation
DISCYR	91-95		Format free,	Date of Reservoir Discovery
DISFLD	97-101		Format free,	Date of Field Discovery
DISMTH	103-104		Format free,	Reservoir Discovery Method

Table A-2 (continued)

File: CANADA.GSM; CANDU.GSM

Data Element	SAS Position	Decimal	FORTRAN Format	Description

SGDVYR	106-110		Format free,	Year Significant Development Drilling Starts
DOMOPR	112-117		Format free,	Dominant Operator as of 1993
WLSPAC	119-123		Format free	Well Spacing (Current Field Rule)
(end of line)				
Record 6				
ACPROD	1-8		Format free,	Estimated Total Production Area (acres)
ACPROV	10-17		Format free,	Maximum Proved Area (acres)
ACPRVD	19-23		Format free,	Date of Maximum Proved Area Estimate
ACDV94	25-32		Format free,	Reservoir Developed Area EOY 1994
OGIP	34-42	.3	Format free,	Reservoir Volumetric Original Gas in Place
CGPR94	44-52	.3	Format free,	Cumulative Gas Production to 1994 (Bcf)
GRSV94	54-62	.3	Format free,	Proved Gas Reserves End of 1994 (Bcf)
RSVCLS	64-66		Format free,	AAPG Reservoir Size Class
GWR94	68-73	.1	Format free,	1994 Gas-Water Ratio
PRSCUR	75-80		Format free,	Current Bottomhole Shut-in Pressure
WATSAC	82-86	.2	Format free,	Current Water Saturation
WATSAB	88-92	.2	Format free,	Abandonment Water Saturation
PRSFLW	94-99		Format free,	Current Bottomhole Flowing Pressure
PRORAT	101-109	.2	Format free,	Proration - Rule for wells/Reservoir
NGLFACT	111-121	.3	Format free	Barrels NGL/MMcf dry gas
(end of line)				
Record 7				
GASPRD82	1-8	.3	Format free,	1982 Gas Production (Bcf)
GASPRD83	9-16	.3	Format free,	1983 Gas Production (Bcf)
GASPRD84	17-24	.3	Format free,	1984 Gas Production (Bcf)
GASPRD85	25-32	.3	Format free,	1985 Gas Production (Bcf)
GASPRD86	33-40	.3	Format free,	1986 Gas Production (Bcf)
GASPRD87	41-48	.3	Format free,	1987 Gas Production (Bcf)
GASPRD88	49-56	.3	Format free,	1988 Gas Production (Bcf)
GASPRD89	57-64	.3	Format free,	1989 Gas Production (Bcf)
GASPRD90	65-72	.3	Format free,	1990 Gas Production (Bcf)
GASPRD91	73-80	.3	Format free,	1991 Gas Production (Bcf)
GASPRD92	81-88	.3	Format free,	1992 Gas Production (Bcf)
GASPRD93	90-97	.3	Format free,	1993 Gas Production (Bcf)
GASPRD94	99-106	.3	Format free	1994 Gas Production (Bcf)
(end of line)				
Record 8				
OILPRD82	1-8		Format free,	1982 Oil Production (Mbbls)
OILPRD83	9-16		Format free,	1983 Oil Production (Mbbls)
OILPRD84	17-24		Format free,	1984 Oil Production (Mbbls)

Table A-2 (continued)

File: CANADA.GSM; CANDU.GSM

Data Element	SAS Position	Decimal	FORTRAN Format	Description
OILPRD85	25-32		Format free,	1985 Oil Production (Mbbls)
OILPRD86	33-40		Format free,	1986 Oil Production (Mbbls)
OILPRD87	41-48		Format free,	1987 Oil Production (Mbbls)

OILPRD88	49-56		Format free,	1988 Oil Production (Mbbls)
OILPRD89	57-64		Format free,	1989 Oil Production (Mbbls)
OILPRD90	65-72		Format free,	1990 Oil Production (Mbbls)
OILPRD91	73-80		Format free,	1991 Oil Production (Mbbls)
OILPRD92	81-88		Format free,	1992 Oil Production (Mbbls)
OILPRD93	90-97		Format free,	1993 Oil Production (Mbbls)
OILPRD94	99-106		Format free	1994 Oil Production (Mbbls)
(end of line)				
Record 9				
NGLPRD82	1-8		Format free,	1982 NGL Production (Mbbls)
NGLPRD83	9-16		Format free,	1983 NGL Production (Mbbls)
NGLPRD84	17-24		Format free,	1984 NGL Production (Mbbls)
NGLPRD85	25-32		Format free,	1985 NGL Production (Mbbls)
NGLPRD86	33-40		Format free,	1986 NGL Production (Mbbls)
NGLPRD87	41-48		Format free,	1987 NGL Production (Mbbls)
NGLPRD88	49-56		Format free,	1988 NGL Production (Mbbls)
NGLPRD89	57-64		Format free,	1989 NGL Production (Mbbls)
NGLPRD90	65-72		Format free,	1990 NGL Production (Mbbls)
NGLPRD91	73-80		Format free,	1991 NGL Production (Mbbls)
NGLPRD92	81-88		Format free,	1992 NGL Production (Mbbls)
NGLPRD93	90-97		Format free,	1993 NGL Production (Mbbls)
NGLPRD94	99-106		Format free	1994 NGL Production (Mbbls)
(end of line)				
Record 10				
TOTWEL82	1-6		Format free,	1982 Total Wells
TOTWEL83	7-12		Format free,	1983 Total Wells
TOTWEL84	13-18		Format free,	1984 Total Wells
TOTWEL85	19-24		Format free,	1985 Total Wells
TOTWEL86	25-30		Format free,	1986 Total Wells
TOTWEL87	31-36		Format free,	1987 Total Wells
TOTWEL88	37-42		Format free,	1988 Total Wells
TOTWEL89	43-48		Format free,	1989 Total Wells
TOTWEL90	49-54		Format free,	1990 Total Wells
TOTWEL91	55-60		Format free,	1991 Total Wells
TOTWEL92	61-66		Format free,	1992 Total Wells
TOTWEL93	68-73		Format free,	1993 Total Wells
TOTWEL94	75-80		Format free	1994 Total Wells
(end of line)				
Record 11				
PRDWEL82	1-6		Format free,	1982 Producing Wells
PRDWEL83	7-12		Format free,	1983 Producing Wells

Table A-2 (continued)

File: CANADA.GSM; CANDU.GSM

Data Element	SAS Position	FORTRAN Decimal	Format	Description
PRDWEL84	13-18		Format free,	1984 Producing Wells
PRDWEL85	19-24		Format free,	1985 Producing Wells
PRDWEL86	25-30		Format free,	1986 Producing Wells
PRDWEL87	31-36		Format free,	1987 Producing Wells
PRDWEL88	37-42		Format free,	1988 Producing Wells

```
    PRDWEL89  43-48             Format free,  1989 Producing Wells
    PRDWEL90  49-54             Format free,  1990 Producing Wells
    PRDWEL91  55-60             Format free,  1991 Producing Wells
    PRDWEL92  61-66             Format free,  1992 Producing Wells
    PRDWEL93  68-73             Format free,  1993 Producing Wells
    PRDWEL94  75-80             Format free   1994 Producing Wells
    (end of line)

Record 12
    PRODHZ82  1-6               Format free,   1982 Producing Horizontal Wells
    PRODHZ83  7-12              Format free,   1983 Producing Horizontal Wells
    PRODHZ84  13-18             Format free,   1984 Producing Horizontal Wells
    PRODHZ85  19-24             Format free,   1985 Producing Horizontal Wells
    PRODHZ86  25-30             Format free,   1986 Producing Horizontal Wells
    PRODHZ87  31-36             Format free,   1987 Producing Horizontal Wells
    PRODHZ88  37-42             Format free,   1988 Producing Horizontal Wells
    PRODHZ89  43-48             Format free,   1989 Producing Horizontal Wells
    PRODHZ90  49-54             Format free,   1990 Producing Horizontal Wells
    PRODHZ91  55-60             Format free,   1991 Producing Horizontal Wells
    PRODHZ92  61-66             Format free,   1992 Producing Horizontal Wells
    PRODHZ93  68-73             Format free,   1993 Producing Horizontal Wells
    PRODHZ94  75-80             Format free    1994 Producing Horizontal Wells
    (end of line)
```

Table A-2 (continued)

File: CANADA.GSM; CANDU.GSM

Data Element	SAS Position	FORTRAN Decimal	FORTRAN Format	Description
Record 13				
TWLSPAC	1-5		Format free,	Target Well Spacing
BPSLOP	7-11	.3	Format free,	Backpressure Exponent
PRSSYS	13-18		Format free,	Operating System Back Pressure (PSIA)
PZSLOP	20-28	.2	Format free,	Slope of Cumulative Production vs. p/z
FLDTYPE	30-31		Format free,	Type of Field
MODULE1	33		Format free	Type Curve Module to use
(end of line)				

Sample Reservoir Data for a Canadian Reservoir GSAMID: 2231C128P001

```
2231C128P001       0 CRANBERRY                               0 CRANBERRY: GILWOOD B POOL      5300          0 C128 100   6
       0        0  1 22   3   57.286 118.603    1 15           7678   4866
   27    14 2.0 320.0     32.960     13.590     32.960 0.128 0.128 0.128  0.45  0.55
  2835 0.640   180  1000  0.005  0.023  0.000 0.354 0.0000030      0   30000 0.0000030 0.00    0.0    0.0   0.0   0.00 0 0
   0.0       0     32.960   0.000 0.000    0.00000      0        0    0.0      0.000 0.00 1977  1977   0 0     0330     320
    1969     1969  1991     1969    14.276     0.565     9.717   2    0.0      0  0.45  0.30      0      0.25      22.659
    0       0       0       0       0       0       0       0       0       0       0       0.035    0.000
       0       0       0       0       0       0       0       0       0       0       0        0        0
       0       0       0       0       0       0       0       0       0       0       0        0        0
     2     2     2     2     2     2     2     2     2     2     2      2      1
     0     0     0     0     0     0     0     0     0     0     1      1      0
     0     0     0     0     0     0     0     0     0     0     0      0      0
  320 0.850    800      0.00 80 1
```

Appendix A - Resource Module Files
(CONTINUED)

These are other summary files (Table A-3 to A-5) created from the discovered Canadian reservoir database. These files do not directly go into GSAM. Instead, they are used for consistency checks.

Table A-3

File; DAT_GSAM.CAN

Data Element	Position	Decimal	Format	Description
GSAMID	1-11		T	Unique GSAM Identification #
GSAMSR	13-14		T	GSAM supply region
RESTYPE	16-19		I	Resource Type
FSCLAS	21-23		I	Field Size Class
PLYCOD	25-28		T	Play Code
PERHOR	30-39	.3	R	Effective Horizontal Permeability
PORTOT	41-45	.3	R	Total Effective Initial Porosity
WATSAT	47-51	.2	R	Initial Water Saturation
DEPTH	53-58		I	Depth
NETPAY	60-69		I	Total Net Pay in Designated Formation
ACPROD	71-80		I	Estimated Total Production Area (Acres)
OGIP	82-91		I	Reservoir Volumetric Original Gas in Place
BHTEMP	93-99		I	Bottomhole Temperature (degrees F)
PRESIN	101-106		I	Initial Reservoir Shut-in Pressure
GASGRV	108-112	.3	R	Specific Gravity of Dry Gas
Z	114-118	.2	R	Compressibility Factor (fraction)
EUR	120-129	.3	R	Estimated Ultimate Recovery

Table A-4

File: LOC_GSAM.CAN

Data Element	Position	Decimal	Format
GSAMID	1-11		T
LEASNAME	13-42		T
EIACOD	44-53		T
FLDNAM	55-79		T
STATE	81-84		I
COUNTY	86-95		I
ADCTYIN	97		T
PLYCOD	99-102		T
BSNCOD	104-106		I
SBPRCOD	108-109		I
LATIN	111-117	.3	R
LONGIT	119-127	.3	R
ONOFFS	129-130		I
DEPTH	132-140		I
(end of line)			

Table A-5

File: PRD_GSAM.CAN

Data Element	Position	Decimal	Format
GSAMID	1-11		T
GASPRD92	13-20	.3	R
GASPRD93	22-29	.3	R
GASPRD94	31-38	.3	R
CGPR94	40-49	.3	R
GRSV94	51-60	.3	R
NGLPRD92	62-69		I
NGLPRD93	71-78		I
NGLPRD94	80-87		I
NGLCRD94	89-98		I
CO2	100-105	.3	R
H2S	107-112	.3	R
N2	114-119	.3	R

Table A-6

This file contains the format of the discovered reservoir properties for APPL.GSM and the undiscovered reservoir properties for UNDISCF.GSM, UNDISCP.GSM, UNDCOLF.GSM, UNDCOLP.GSM, UNDTGTF.GSM, UNDTGTP.GSM, UNDOFFF.GSM (UNDATL.GSM and UNDGOME.GSM), UNDCCN.GSM, UNDTCN.GSM, UNDCAN.GSM, and UNDCHYP.GSM.

```
01126728F005    5855.   5.0     .010  .042  .300  5900.   2764. 178.   .0000  .0000  .0000          8     37  .650    0                         0.
01126728F006    8432.   6.9     .010  .042  .300  5900.   2764. 178.   .0000  .0000  .0000          4     37  .650    0                         0.
01126728F007   12142.   9.6     .010  .042  .300  5900.   2764. 178.   .0000  .0000  .0000          2     37  .650    0                         0.
01126728F008   17486.  13.4     .010  .042  .300  5900.   2764. 178.   .0000  .0000  .0000          2     37  .650    0                         0.
01126728F009   25182.  18.6     .010  .042  .300  5900.   2764. 178.   .0000  .0000  .0000          0     37  .650    0                         0.
01126728F010   36264.  25.8     .010  .042  .300  5900.   2764. 178.   .0000  .0000  .0000          0     37  .650    0                         0.
01126728F011   52223.  35.9     .010  .042  .300  5900.   2764. 178.   .0000  .0000  .0000          0     37  .650    0                         0.
01126728F012   75206.  49.8     .010  .042  .300  5900.   2764. 178.   .0000  .0000  .0000          1     37  .650    0                         0.
01126728F013  108304.  69.2     .010  .042  .300  5900.   2764. 178.   .0000  .0000  .0000          0     37  .650    0                         0.
01126728F014  155968.  96.1     .010  .042  .300  5900.   2764. 178.   .0000  .0000  .0000          0     37  .650    0                         0.
01126728F015  224608. 133.5     .010  .042  .300  5900.   2764. 178.   .0000  .0000  .0000          0     37  .650    0                         0.
01126728F016  323457. 185.4     .010  .042  .300  5900.   2764. 178.   .0000  .0000  .0000          0     37  .650    0                         0.
01126728F017  465807. 257.4     .010  .042  .300  5900.   2764. 178.   .0000  .0000  .0000          0     37  .650    0                         0.
01126729F005    6968.   5.0     .010  .042  .300  4500.   2113. 150.   .0000  .0000  .0000          2     31  .650    0                         0.
01126729F006   10035.   6.9     .010  .042  .300  4500.   2113. 150.   .0000  .0000  .0000          2     31  .650    0                         0.
01126729F007   14451.   9.6     .010  .042  .300  4500.   2113. 150.   .0000  .0000  .0000          1     31  .650    0                         0.
01126729F008   20810.  13.4     .010  .042  .300  4500.   2113. 150.   .0000  .0000  .0000          0     31  .650    0                         0.
01126729F009   29969.  18.6     .010  .042  .300  4500.   2113. 150.   .0000  .0000  .0000          0     31  .650    0                         0.
01126729F010   43158.  25.8     .010  .042  .300  4500.   2113. 150.   .0000  .0000  .0000          1     31  .650    0                         0.
01126729F011   62151.  35.9     .010  .042  .300  4500.   2113. 150.   .0000  .0000  .0000          0     31  .650    0                         0.
01126729F012   89503.  49.8     .010  .042  .300  4500.   2113. 150.   .0000  .0000  .0000          0     31  .650    0                         0.
01126729F013  128893.  69.2     .010  .042  .300  4500.   2113. 150.   .0000  .0000  .0000          0     31  .650    0                         0.
01126729F014  185618.  96.1     .010  .042  .300  4500.   2113. 150.   .0000  .0000  .0000          0     31  .650    0                         0.
01126729F015  267307. 133.5     .010  .042  .300  4500.   2113. 150.   .0000  .0000  .0000          0     31  .650    0                         0.
01126729F016  384946. 185.4     .010  .042  .300  4500.   2113. 150.   .0000  .0000  .0000          0     31  .650    0                         0.
01126729F017  554358. 257.4     .010  .042  .300  4500.   2113. 150.   .0000  .0000  .0000          0     31  .650    0                         0.
01126730F005    5795.   5.0     .010  .042  .300  6000.   2811. 180.   .0000  .0000  .0000          0     31  .650    0                         0.
01126730F006    8345.   6.9     .010  .042  .300  6000.   2811. 180.   .0000  .0000  .0000          1     31  .650    0                         0.
01126730F007   12018.   9.6     .010  .042  .300  6000.   2811. 180.   .0000  .0000  .0000          0     31  .650    0                         0.
01126730F008   17307.  13.4     .010  .042  .300  6000.   2811. 180.   .0000  .0000  .0000          0     31  .650    0                         0.
01126730F009   24924.  18.6     .010  .042  .300  6000.   2811. 180.   .0000  .0000  .0000          0     31  .650    0                         0.
01126730F010   35893.  25.8     .010  .042  .300  6000.   2811. 180.   .0000  .0000  .0000          0     31  .650    0                         0.
01126730F011   51689.  35.9     .010  .042  .300  6000.   2811. 180.   .0000  .0000  .0000          0     31  .650    0                         0.
01126730F012   74437.  49.8     .010  .042  .300  6000.   2811. 180.   .0000  .0000  .0000          0     31  .650    0                         0.
01126730F013  107196.  69.2     .010  .042  .300  6000.   2811. 180.   .0000  .0000  .0000          0     31  .650    0                         0.
01126730F014  154372.  96.1     .010  .042  .300  6000.   2811. 180.   .0000  .0000  .0000          0     31  .650    0                         0.
01126730F015  222310. 133.5     .010  .042  .300  6000.   2811. 180.   .0000  .0000  .0000          0     31  .650    0                         0.
01126730F016  320148. 185.4     .010  .042  .300  6000.   2811. 180.   .0000  .0000  .0000          0     31  .650    0                         0.
01126730F017  461042. 257.4     .010  .042  .300  6000.   2811. 180.   .0000  .0000  .0000          0     31  .650    0                         0.
01126733F005    6870.   5.0     .010  .042  .300  4600.   2158. 152.   .0000  .0000  .0000          2     37  .650    0                         0.
01126733F006    9893.   6.9     .010  .042  .300  4600.   2158. 152.   .0000  .0000  .0000          2     37  .650    0                         0.
01126733F007   14247.   9.6     .010  .042  .300  4600.   2158. 152.   .0000  .0000  .0000          2     37  .650    0                         0.
01126733F008   20517.  13.4     .010  .042  .300  4600.   2158. 152.   .0000  .0000  .0000          0     37  .650    0                         0.
01126733F009   29546.  18.6     .010  .042  .300  4600.   2158. 152.   .0000  .0000  .0000          0     37  .650    0                         0.
01126733F010   42550.  25.8     .010  .042  .300  4600.   2158. 152.   .0000  .0000  .0000          1     37  .650    0                         0.
01126733F011   61275.  35.9     .010  .042  .300  4600.   2158. 152.   .0000  .0000  .0000          1     37  .650    0                         0.
01126733F012   88242.  49.8     .010  .042  .300  4600.   2158. 152.   .0000  .0000  .0000          0     37  .650    0                         0.
01126733F013  127077.  69.2     .010  .042  .300  4600.   2158. 152.   .0000  .0000  .0000          0     37  .650    0                         0.
01126733F014  183002.  96.1     .010  .042  .300  4600.   2158. 152.   .0000  .0000  .0000          0     37  .650    0                         0.
01126733F015  263540. 133.5     .010  .042  .300  4600.   2158. 152.   .0000  .0000  .0000          0     37  .650    0                         0.
01126733F016  379522. 185.4     .010  .042  .300  4600.   2158. 152.   .0000  .0000  .0000          0     37  .650    0                         0.
01126733F017  546547. 257.4     .010  .042  .300  4600.   2158. 152.   .0000  .0000  .0000          0     37  .650    0                         0.
01126734F005    6190.   5.0     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          4     37  .650    0                         0.
01126734F006    8914.   6.9     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          2     37  .650    0                         0.
01126734F007   12836.   9.6     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          1     37  .650    0                         0.
01126734F008   18486.  13.4     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     37  .650    0                         0.
01126734F009   26621.  18.6     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     37  .650    0                         0.
01126734F010   38337.  25.8     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          1     37  .650    0                         0.
01126734F011   55209.  35.9     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     37  .650    0                         0.
01126734F012   79505.  49.8     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     37  .650    0                         0.
01126734F013  114495.  69.2     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     37  .650    0                         0.
01126734F014  164884.  96.1     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     37  .650    0                         0.
01126734F015  237447. 133.5     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     37  .650    0                         0.
01126734F016  341946. 185.4     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     37  .650    0                         0.
01126734F017  492434. 257.4     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     37  .650    0                         0.
01126735F005    6190.   5.0     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          1     31  .650    0                         0.
01126735F006    8914.   6.9     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          1     31  .650    0                         0.
01126735F007   12836.   9.6     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     31  .650    0                         0.
01126735F008   18486.  13.4     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          1     31  .650    0                         0.
01126735F009   26621.  18.6     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     31  .650    0                         0.
01126735F010   38337.  25.8     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     31  .650    0                         0.
01126735F011   55209.  35.9     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     31  .650    0                         0.
01126735F012   79505.  49.8     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     31  .650    0                         0.
01126735F013  114495.  69.2     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     31  .650    0                         0.
01126735F014  164884.  96.1     .010  .042  .300  5400.   2530. 168.   .0000  .0000  .0000          0     31  .650    0                         0.
```

Description of File: APPL.GSM, UNDXXXX.GSM

This is the generic format of the files of the kind UNDXXXX.GSM and APPL.GSM, listed on page A-15, where the “XXXX” symbol stands for a string of letters designating the country, the predominant resource type, and the federal/private land flag of the reservoir that the file describes. (for example, UNDTGTF.GSM contains tight reservoirs in the United States, located on federal land).

Data Element	Description	Format
1	12-digit GSAMID	A12
2	Area (acre)	T15, F7.0
3	Net pay (ft)	1x, F5.1
4	Permeability (md)	1x, F8.3
5	Porosity (fraction)	1x, F5.3
6	Water saturation (fraction)	1x, F5.3
7	Depth (ft)	1x, F6.0
8	Initial pressure (psia)	1x, F6.0
9	Bottom hole temperature (degrees F)	1x, F4.0
10	CO_2 concentration (fraction)	1x, F6.4
11	N_2 concentration (fraction)	1x, F6.4
12	H_2S concentration (fraction)	1x, F6.4
13	No. of reservoirs accumulations	T92, F9.0
14	Four-digit state code	1x, I6
15	Gas gravity	1x, F5.3
16	Start year of the reservoir (for Appalachian reservoir used for history check calculation; for all others is zero)	1x, I4
17	Water Depth	t139,f6.0

Table A-7

Dictionary for State/District Codes: (4 digit state code; the last two digits are used to distinguish different regions within a state).

State/District Code	State/District
0100	ALABAMA FED. OFFSHORE
0105	ALABAMA STATE OFFSHORE
0110	ALABAMA ONSHORE
5000	ALASKA SOUTH FED. OFFSHORE
5005	ALASKA SOUTH STATE OFFSHORE
5010	ALASKA SOUTH ONSHORE
5050	ALASKA NORTH ONSHORE
2	ARIZONA
0310	ARKANSAS SOUTH
0350	ARKANSAS NORTH
0400	CALIFORNIA FED. OFFSHORE
0405	CALIFORNIA STATE OFFSHORE
0410	CALIFORNIA CENTRAL VALLEY
0450	CALIFORNIA COASTAL
0490	CALIFORNIA LOS ANGELES BASIN
5	COLORADO
0900	FLORIDA FED. OFFSHORE
0910	FLORIDA ONSHORE
12	ILLINOIS
13	INDIANA
15	KANSAS
16	KENTUCKY
1700	LOUISIANA FED. OFFSHORE
1705	LOUISIANA STATE OFFSHORE
1710	LOUISIANA SOUTH
1750	LOUISIANA NORTH
19	MARYLAND
21	MICHIGAN
2300	MISSISSIPPI FED. OFFSHORE
2310	MISSISSIPPI ONSHORE
24	MISSOURI
25	MONTANA
26	NEBRASKA
27	NEVADA
3010	NEW MEXICO SOUTHEAST
3050	NEW MEXICO NORTHWEST
31	NEW YORK
33	NORTH DAKOTA
34	OHIO
3510	OKLAHOMA SOUTHWEST
3520	OKLAHOMA SOUTHEAST
3530	OKLAHOMA NORTHEAST
3540	OKLAHOMA NORTH CENTRAL

Table A-7 (continued)

State/District Code	State/District
3550	OKLAHOMA NORTHWEST
36	OREGON
37	PENNSYLVANIA
40	SOUTH DAKOTA
41	TENNESSEE
4200	TEXAS FED. OFFSHORE
4205	TEXAS STATE OFFSHORE
4210	TEXAS RRC DISTRICT 1
4220	TEXAS RRC DISTRICT 2
4230	TEXAS RRC DISTRICT 3
4240	TEXAS RRC DISTRICT 4
4250	TEXAS RRC DISTRICT 5
4260	TEXAS RRC DISTRICT 6
4270	TEXAS RRC DISTRICT 7B
4275	TEXAS RRC DISTRICT 7C
4280	TEXAS RRC DISTRICT 8
4285	TEXAS RRC DISTRICT 8A
4290	TEXAS RRC DISTRICT 9
4295	TEXAS RRC DISTRICT 10
43	UTAH
45	VIRGINIA
46	WASHINGTON
47	WEST VIRGINIA
49	WYOMING
71	ALBERTA, CANADA
72	BRITISH COLUMBIA, CANADA
73	MANITOBA, CANADA
81	SASKATCHEWAN, CANADA

Table A-8

Dictionary for GSAM Supply Regions:

Region Code	GSAM Region Code	Region
01	AP	APPALACHIA
02	MF	MAFLA ONSHORE
03	MW	MID-WEST
04	AET	ARKLA-EAST TEXAS
05	SL	SOUTHERN LOUISIANA
06	TGC	TEXAS GULF COAST
07	P	PERMIAN
08	MC	MID-CONTINENT
09	SJ	SAN JUAN
10	RF	ROCKIES FORELAND
11	WL	WILLISTON
12	PON	PACIFIC ONSHORE
13	AOF	ATLANTIC OFFSHORE
14	GME	GULF OF MEXICO-EAST
15	NP	NORPHLET
16	GMC	GULF OF MEXICO-CNTR
17	GMW	GULF OF MEXICO-WEST
18	POF	PACIFIC OFFSHORE
19	NA	NORTH ALASKA
20	MD	MACKENZIE DELTA
21	--	Not used currently
22	ALB	ALBERTA (including other provinces in Canada)
23	BC	BRITISH COLUMBIA
24	EC	EASTERN CANADA

Table A-9

Dictionary for Module Types (the “Module1” variable of the database):

Module Code	Module Type
1	CONVENTIONAL
2	TIGHT SAND
3	DUAL POROSITY (RADIAL FLOW; naturally fractured reservoirs)
4	DUAL POROSITY (LINEAR FLOW; naturally fractured reservoirs with induced hydraulic fractures)
5	WATER DRIVE
6	COAL/SHALE
7	ANALYZED CONVENTIONAL

Table A-10

Dictionary for Lithology Types:

Lithology Code	Lithology Type
0	NOT AVAILABLE
5	CONGLOMERATE
10	SANDSTONE
15	SILTSTONE
20	SHALE
30	CARBONATE
35	LIMESTONE
40	DOLOMITE
45	CHERT (SPECIALIZED DOLOMITE)
50	CHALK
55	ANHYDRITE
60	GRANITE WASH
70	COAL
80	SCHIST
85	QUARTZITE
90	IGNEOUS
95	VOLCANICS

Table A-11

Dictionary for Status:

Status Code	Status
1	UNDISCOVERED
2	DISCOVERED, UNDEVELOPED
3	DEVELOPED

Table A-12

Dictionary for Trap Type:

Trap Type Code	Trap Type
0	NOT RELEVANT
1	STRUCTURE
2	FAULT
3	UNCONFORMITY
4	STRATIGRAPHIC DOMINANT TRAPPING MECHANISM
6	COMBINATION

Table A-13

Dictionary for Dominant Drive Type:

Dominant Drive Code	Dominant Drive Type
0	UNKNOWN
1	PRESSURE DEPLETION (COMPACTION)
2	WATER DRIVE
2	PARTIAL WATER DRIVE
3	SOLUTION (DISSOLVED GAS)
4	GAS CAP EXPANSION
5	GRAVITY DRAINAGE
6	NO STRONG (INITIAL BY LIFT)

Table A-14

Dictionary for Type of Field:

Field Type Code	Field Type
1	OIL
2	GAS
3	BOTH
4	NONE

Table A-15

Dictionary for field size class:

Field Class Size	Average recoverable resource (BCF)	Average OGIP (BCF)
5	4.5	6.62
6	9	13.24
7	18	26.47
8	36	52.94
9	72	105.88
10	144	211.76
11	288	423.53
12	576	847.06
13	1152	1694.12
14	2304	3388.23
15	4608	6776.47
16	9216	13552.94
17	18432	27105.88

APPENDIX B
RESERVOIR PERFORMANCE MODULE FILES

CONTENTS

[1] These are auxiliary files which may be printed by specifying in REGIONS.DAT file, but should only be used for consistency checks, as they will be prohibitively large in a full RP run

[2] These binary files are always created by the RP Module and contain production and pressure information. They may be used to greatly speed up runs when technology parameters do not change.

Table B-1
Input Data File: AFE.DAT (Location: \GSAM\RESVPERF\DATA)
This file shows the percentages of investment in normal AFE categories.

```
C*** AFE Proportions
    Category                        % of Total
C*****************************        C*****
Contractor Charges                    30.7
Road & Site Prep                      4.4
Transportation                        2.4
Fuel                                  0.6
Mud & Additives                       5.8
Drillsite Logs & Monitoring           1.1
Other Physical Test                   0.6
Logs & Wireline Evaluations           3.7
Wellsite Data Services                0.1
Directional Drilling Services         1.6
Perforating                           1.4
Formation Treatment                   4.4
Cement & Services                     4.8
Casing & Tubing                       13.7
Special Tool Rentals                  2.7
Drill Bits & Reamers                  2.3
Wellhead Equipment                    1.8
Other Equipment & Supplies            3.0
Plugging                              1.3
Supervision & Overhead                5.6
Other Expenditures                    7.1
```

Explanation

The AFE.DAT cost file contains the authorization for expenditure charges for a producer. These percentages are taken from various sources including Joint Association Survey publications, the 1997 Well Cost Study by Petroleum Services Association of Canada (PSAC), and other ICF statistical estimates.

Intended Uses

This file is not currently used in running the RP Module. It does, however, provide the user with the structure of costing for a completed producing well.

Table B-2

Input Data File: COST.DAT (Location: \GSAM\RESVPERF\DATA)
This is a costing file for the Reservoir Performance Module. See below for a complete description of the parameters.

```
C*** Discount  Rate(%)
10.0
C*** Number of Technology Cases
2
C*** Name of Case One
Current Technology
C*** Exploratory Well Costs Factor (multiplied with DWC to get EWC)
1.2
C*** Lease Bonus Cost Factor (multiplied with total revenues to get Lease Bonus)
0.005
C*** G&G Factor (Portion of EWC that is G&G costs)
0.05
C*** Development Dry Hole Costs as % of Total Development Well Costs (%)
70.0
C*** Percent Exploratory Well Cost Tangible (%)
25.0
C*** Percent Development Well Cost Tangible (%)
40.0
C*** Percent Facilities Cost Tangible (%)
100.00
C*** Environmental Capital Cost Multiplier (scaler of Facilities)
0.10
C*** G&A Expense Multiplier (scalar)
0.25
C*** G&A Capital Multiplier (scalar)
0.10
C*** Number of Regions (Excluding Default - 99)
21
C*** Development Well Cost (Function of Well Depth)
  1      27.068800   4.7098399e-2 -2.547277e-6  1.18087525e-10 1
  2      59.25750    5.948449e-2  -3.611307e-6  3.975062e-10   1
  3      34.16550    9.042406e-2  -3.209655e-7  9.641927e-10   1
  4      46.55651   -1.905254e-2   1.430119e-5 -3.249473e-10   1
  5      299.0901    1.280414e-1  -1.890103e-5  1.784502e-9    1
  6      21.7314     1.848788e-3   8.429423e-6  2.456291e-10   1
  7      35.65350    8.322879e-2  -1.829027e-5  1.632399e-9    1
  8      78.07079    1.775666e-2   1.481531e-6  3.708914e-10   1
  9      54.40730    5.387449e-2  -7.106254e-6  1.339750e-9    1
 10      47.21670    5.970161e-2  -6.851103e-6  7.622377e-10   1
 11      17.8909     3.34051e-2    1.13753e-6   0.0            1
 12      200.0000   -2.418747e-2   2.630253e-5 -6.552413e-10   1
 13      1715.511    3.880667e-1  -2.868301e-5  1.556441e-9    1
 14      59.25750    5.948449e-2  -3.611307e-6  3.975062e-10   1
 15      1715.511    3.880667e-1  -2.868301e-5  1.556441e-9    1
 16      1715.511    3.880667e-1  -2.868301e-5  1.556441e-9    1
 17      2239.81    -4.69329e-2    2.14607e-5   3.03929e-10    1
 18      1000.0      1.848788e-3   8.429423e-6  2.456291e-10   1
 19       300.0      1.848788e-3   8.429423e-6  2.456291e-10   1
 22      17.8909     3.34051e-2    1.13753e-6   0.0            1

 23      17.8909     3.34051e-2    1.13753e-6   0.0            1
 99      78.07079    1.775666e-2   1.481531e-6  3.708914e-10   1
c regions
0
C**Environmental Costs
 99   0       0       0       0       0       0       0       0       0
C*** Number of Regions For Facilities Well Cost (K$)
15
Region#  # of Depth Steps
01             1
Max Depth      $/Well   $/Well/MCF-D
---------    --------- --------------
160000         261.8       26.18
Region#  # of Depth Steps
02             5
Max Depth      $/Well   $/Well/MCF-D
```

```
---------    --------- -------------
2000         19707.00      0.00
4000         22278.00     -0.91
8000         23280.00     20.72
12000        20478.00     39.04
16000        34399.00     11.02
Region#  # of Depth Steps
03              5
Max Depth      $/Well   $/Well/MCF-D
---------    --------- -------------
2000         19438.80      1.36
4000         18777.00     23.51
8000         20266.80     38.23
12000        31581.30     15.85
16000        35252.20     11.44
Region#  # of Depth Steps
04              5
Max Depth      $/Well   $/Well/MCF-D
---------    --------- -------------
2000         19707.00      0.00
4000         22278.00     -0.91
8000         23280.00     20.72
12000        20478.00     39.04
16000        34399.00     11.02
Region#  # of Depth Steps
05              5
Max Depth      $/Well   $/Well/MCF-D
---------    --------- -------------
2000         19707.00      0.00
4000         18552.00     23.09
8000         19976.00     40.93
12000        33265.00     11.86
16000        35531.00     12.82
Region#  # of Depth Steps
06              4
Max Depth      $/Well   $/Well/MCF-D
---------    --------- -------------
2000         19040.00      0.00
4000         17910.00     22.61
8000         19196.00     40.62
12000        32375.00     11.80
Region#  # of Depth Steps
07              5
Max Depth      $/Well   $/Well/MCF-D
---------    --------- -------------
2000         16946.00      0.00
4000         15815.00     22.61
8000         17091.00     46.53
12000        21951.00     42.08
16000        33792.00     11.64
Region#  # of Depth Steps
08              5
Max Depth      $/Well   $/Well/MCF-D
---------    --------- -------------
2000         19445.00      9.04
4000         19811.00     15.97
8000         32095.00     20.81
12000        24812.00     36.70
16000        36219.00     11.20
Region#  # of Depth Steps
09              4
Max Depth      $/Well   $/Well/MCF-D
---------    --------- -------------
2000         21611.00      0.00
4000         19141.00     54.50
8000         21663.00     53.48
12000        29647.00     36.65
Region#  # of Depth Steps
10             4
Max Depth      $/Well   $/Well/MCF-D
---------    --------- -------------
2000         21611.00      0.00
4000         19141.00     54.50
8000         21663.00     53.48
```

```
12000         29647.00      36.65
Region#  # of Depth Steps
11             4
Max Depth      $/Well   $/Well/MCF-D
---------    --------- --------------
2000          21611.00       0.00
4000          19141.00      54.50
8000          21663.00      53.48
12000         29647.00      36.65
Region#  # of Depth Steps
12             5
Max Depth      $/Well   $/Well/MCF-D
---------    --------- --------------
2000          19445.00       9.04
4000          19811.00      15.97
8000          32095.00      20.81
12000         24812.00      36.70
16000         36219.00      11.20
Region#  # of Depth Steps
15             5
Max Depth      $/Well   $/Well/MCF-D
---------    --------- --------------
2000          19707.00       0.00
4000          20415.00      11.09
8000          21628.00      30.82
12000         26871.50      25.45
16000         34965.00      11.92
Region#  # of Depth Steps
16             5
Max Depth      $/Well   $/Well/MCF-D
---------    --------- --------------
2000          19707.00       0.00
4000          18552.00      23.09
8000          19976.00      40.93
12000         33265.00      11.86
16000         35531.00      12.82
Region#  # of Depth Steps
17             4
Max Depth      $/Well   $/Well/MCF-D
---------    --------- --------------
2000          19040.00       0.00
4000          17910.00      22.61
8000          19196.00      40.62
12000         32375.00      11.80
Region#  # of Depth Steps
99             5
Max Depth      $/Well   $/Well/MCF-D
---------    --------- --------------
2000        19409     1.51
4000        18388    26.22
8000        19932    40.18
12000       32815    13.29
16000       35347    11.48
C*** STMFAC(ITECH) Value, fraction
0.60
C*** Compression Cost ($/BHP) !Assumption is that single stage compressor is used
1200
C*** Variable O&M Water ($/Barrel)
0.25
C*** Variable O&M Gas ($/Mcf)+Incremental of per 1000 feet depth
0.005   0.0
C*** Compressor O&M ($/Mcf)
0.05
C*** Annual Fixed O&M Well Cost (function of well depth)
C*** Number of Regions (Excluding Default - 99)
12
C-  C-    Region and Number of Steps
01  1                                      Marginal
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 16000.0         8364.1        2.00
C-  C-    Region and Number of Steps
04  1                                      Marginal
Max. Depth       $/Well      $/(Well-ft)
```

```
C---------     C---------    C----------
 16000.0        6154.1       2.38
C-  C-    Region and Number of Steps
05  1                                          Marginal
Max. Depth      $/Well      $/(Well-ft)
C---------     C---------    C----------
 16000.0        8800.7       1.91
C-  C-    Region and Number of Steps
06  1                                          Marginal
Max. Depth      $/Well      $/(Well-ft)
C---------     C---------    C----------
 12000.0        6720.6       2.12
C-  C-    Region and Number of Steps
07  1                                          Marginal
Max. Depth      $/Well      $/(Well-ft)
C---------     C---------    C----------
 16000.0        6647.5       1.87
C-  C-    Region and Number of Steps
08  1                                          Marginal
Max. Depth      $/Well      $/(Well-ft)
C---------     C---------    C----------
 16000.0        7950.2       2.04
C-  C-    Region and Number of Steps
10  1                                          Marginal
Max. Depth      $/Well      $/(Well-ft)
C---------     C---------    C----------
 12000.0       10821.0       2.25
C-  C-    Region and Number of Steps
13  1                                          Marginal
Max. Depth      $/Well      $/(Well-ft)
C---------     C---------    C----------
 16000.0      250757.0       0.00
C-  C-    Region and Number of Steps
14  1                                          Marginal
Max. Depth      $/Well      $/(Well-ft)
C---------     C---------    C----------
 16000.0        8800.7       1.91
C-  C-    Region and Number of Steps
15  1                                          Marginal
Max. Depth      $/Well      $/(Well-ft)
C---------     C---------    C----------
 16000.0        7477.4       2.14
C-  C-    Region and Number of Steps
16  1                                          Marginal
Max. Depth      $/Well      $/(Well-ft)
C---------     C---------    C----------
 16000.0        8807.0       1.91
C-  C-    Region and Number of Steps
17  1                                          Marginal
Max. Depth      $/Well      $/(Well-ft)
C---------     C---------    C----------
 16000.0        6720.6       2.12
C-  C-    Region and Number of Steps
99  1                                          Marginal
Max. Depth      $/Well      $/(Well-ft)
C---------     C---------    C----------
 16000.0        8364.0       2.00
```

Explanation

The 10% **discount rate** reflects normal assumptions about inflation and the opportunity cost of capital, and may be changed as desired.

The **number of technology cases** is set in the "Number of Technology Cases" line. This number may be either 1 or 2. Be sure that each technology case is named in the next line.

The **exploratory well cost factor** describes how the inherent characteristics of exploratory drilling make it more expensive than development drilling. NOTE however that the RP Module does not currently model the drilling of exploration wells (it is done in the E&P Module), so that this factor is NOT currently used in the RP model. A similar parameter is specified in the E&P Module's DRL_CST.SPC file, and is used in the E&P Module.

The **lease bonus cost factor** is assumed to be a fraction of the total revenue that could be generated from the reservoir, and the lease bonus cost is calculated by multiplying the factor with the total collected revenue.

The **G&G costs** are a fraction of the exploratory well costs. However, because the RP Module does not currently model exploratory drilling this factor is not presently employed.

Based on various reports, like the PSAC Well Cost Study, the **dry hole cost** is on average 70% of the total cost for a completed development well producer.

The **tangible and intangible percentages** are industry averages. It is assumed that surface facilities are 100% tangible for tax accounting purposes.

The **environmental capital cost multiplier** is a percentage of the facilities cost. It is the base environmental cost. A 10% factor, for example, designates surface facilities installed that handle gas stream (including impurities), water production, etc. at 10% of the total surface facilities cost. The incremental environmental compliance costs in the E&P Module or the RP Module are on top of this base cost. The E&P compliance costs are incremental and are applied in the E&P Module through various files. The RP compliance costs (described below) are incremental but are constant through time.

The regional **development well cost** table contains entries that correspond to the coefficients of a polynomial regression equation that is the best fit of the historical cost vs. depth data from the 1997 JAS Survey. As such, these entries should not be altered unless a similar procedure is undertaken. The number of regions, excluding the default values (#99), must be set before these values. The drilling cost coefficient values appear in 4 columns and are in thousand dollars as a function of depth. The first cost column is the intercept and the next three are the coefficients of x^a where x is depth, in feet. The development drilling cost calculation is demonstrated in the following example:

For region #1 the cost columns are: 27.068800 4.7098399e-2 -2.547277e-6 1.18087525e-10
So that: cost = $27.07 + (4.71e^{-2})x + (-2.55e^{-6})x^2 + (1.18e^{-10})x^3$
If depth x = 1000 feet then:

$$\text{cost} = 27.07 + (4.71e^{-2})(1000) + (-2.55e^{-6})(1000)^2 + (1.18e^{-10})(1000)^3 = 71.738$$

From these calculations, the cost of a development well in region #1 (Appalachia) at a depth of 1000 feet is $71,738. The column on the far right is a multiplier (of the development well drilling cost) for horizontal or vertical technology (1.3 means that horizontal wells are 130% as costly as vertical wells). The vertical well cost file should normally have 1's in this column. Note that this factor can be used for modeling any other technology such as costlier drilling mud (synthetic muds, sour formations, etc.)

Environmental costs in general are specified in the E&P module. However, if the user wants to use environmental compliance costs from start of the run in the RP module, then they can be specified. The number of regions that have environmental costs must be specified. The entries are as follows :

Data Element 1:	GSAM supply region indicator
Data Element 2:	Existing well environmental tangible capital cost (K$/Well), incremental
Data Element 3:	Existing well environmental intangible capital cost (K$/Well), incremental
Data Element 4:	Existing well environmental operating cost (K$/Well), incremental
Data Element 5:	New well environmental tangible capital cost (K$/Well), incremental
Data Element 6:	New well environmental intangible capital cost (K$/Well), incremental
Data Element 7:	New well environmental operating cost (K$/Well), incremental
Data Element 8:	Incremental environmental cost related to drilling, %/ft

Data Element 9: Incremental environmental cost related to gas production handling (impurities), $/MCF
Data Element 10: Incremental environmental cost related to associated water production, $/BBL

The year these environmental costs would be applicable is specified in RUNSET.DAT file.

The Facilities Well Cost section of the COST.DAT file consists of two sets of data. The first set has the number of regions that will have facilities costs reported. The second set has the coefficients for calculating the costs for each region with the region number and the number of depth steps used. The facilities well cost is designed based on the gas flow rate from the well. The file currently has 10,000 Mcf/day as a “maximum” flow rate. This structure allows for facilities well costs to be applied in steps, so that the facilities costs could vary with different production rates.
The source for updating the Facilities Well Cost section of the COST.DAT is the “Cost and Indices for Domestic Oil and Gas Field Equipment and Production Operations 1995 through 1998” (EIA, 1999).

Stimulation Factor (STMFAC) is defined as design factor in calculating fracture cost. A value of 0.60 means that if the reservoir is fractured for a 500 ft fracture half length, then the actual cost would be for a fracture of 500/0.60 = 833.33 ft.

The **compression costs** assume that single stage compressor is used.

Variable O&M gas cost depends upon how much gas is handled and represents electricity use and cost factors such as more trips to the fields, etc. Most of the variable O&M is the **compressor O&M**. The fixed O&M values have also been calculated from the "Costs and Indices for Domestic Oil and Gas Field Equipment and Production Operations" August 1996 report in similar fashion to the facilities well cost. It is a function of well depth, and again, the regions must be set.

The Fixed O&M costs section of the COST.DAT file consists of two sets of data. The first set has the number of regions that will have fixed O&M costs reported. The second set has the coefficients for calculating the costs for each region with the region number and the number of depth steps used. The fixed O&M cost is designed based on the depth of the well. Currently the fixed O&M cost section of the COST.DAT file uses only one depth step at a maximum depth of 16,000 ft.
The source for updating the Fixed O&M Cost section of the COST.DAT is the “Cost and Indices for Domestic Oil and Gas Field Equipment and Production Operations 1995 through 1998” (EIA, 1999).

The COST.DAT file is normally set up for two technologies: current and advanced. Under **advanced technology** several assumptions have been made, and may be changed if desired. Facilities well costs are improved by 20%, drilling costs by 10%, and compressor O&M by 1%.

Intended Uses of COST.DAT

COST.DAT is intended to be used for changing the costing parameters of the Reservoir Performance Module, impacting the economics of a reservoir, and the subsequent decisions in the E&P Module. Many of the parameters may be altered for sensitivity analysis. In sensitivity analysis, although the cases must be named current and advanced, this does not necessarily mean that, for instance, advanced must model an advanced technology. A user could, if desired, change the horizontal drilling cost in one region, and have all the information the same in the COST.DAT file, to model sensitivity to the cost of drilling a horizontal well.

A note on the functioning of the RP and E&P Modules: The costs and financial information used in the RP Module (such as NPV of investment, NPV of expenses, NPV of drilling costs, NPV of non-drilling costs, etc.) are stored in the .DEC file. The E&P Module performs a linear interpolation/extrapolation of these numbers at a specified gas price. This is possible because the RP uses flat gas prices of $2/Mcf and $5/Mcf, using $2 for the NPV calculations. These computations are then updated at the specified gas price track in the E&P Module.

Table B-3

Input Data File: GEOLOGY.DAT (Location: \GSAM\RESVPERF\DATA)
This file contains reservoir property distribution by pay grade.

```
C*** Number of Resource Types  (Excluding Default)
2
C*** Pay Grade Parameters for Resource Type (Last is Default - 99)
Resource   P.G.   Acreage   Porosity    Netpay   H2O Saturation  Permeability
  type
   1       1      0.20      1.0        1.2000      1.0             1.5
   1       2      0.50      1.0         1.0        1.0             1.0
   1       3      0.30      1.0        0.850       1.0            0.75
   6       1      0.30      1.0        1.0000      1.0             1.0
   6       2      0.40      1.0        1.0000      1.0             1.0
   6       3      0.30      1.0        1.0000      1.0             1.0
   99      1      0.20      1.0        1.5000      1.0             2.0
   99      2      0.50      1.0         1.0        1.0             1.0
   99      3      0.30      1.0        0.6667      1.0            0.50
```

Explanation

Set the number of resource types to more than 1 if pay grade distribution (heterogeneity) within the resource is to be modeled. In this example, conventional and coalbed reservoirs have different characteristics. Note that **(**Area*Porosity*Netpay*H2O Saturation For PAY GRADE 1 + Area*Porosity*Netpay*H2O Saturation For PAY GRADE 2 + Area*Porosity*Netpay*H_2O Saturation For PAY GRADE 3**)** = 1.0. If this equation does not hold, the model will not function. The equation means that all three pay grades volumetrically contain the same amount of gas as would be contained in the reservoir without pay grade property variation. Permeability variation for conventional is assumed such that pay grade 1 is 50% better than the average specified permeability (*.GSM file value) and pay grade 3 is 25% worse. For tight, water drive, and naturally fractured reservoirs in the above file, the pay grade distribution is specified as per the default value (i.e., #99).

Intended Uses of GEOLOGY.DAT

Use GEOLOGY.DAT to characterize a resource type's pay grade properties. This file can be used to study the sensitivity of reservoir heterogeneity. It should be noted that all discovered and undiscovered reservoirs in a given resource type would have the same pay grade distribution.

Table B-4

Input Data File: PLY_DFN.SPC (Location: \GSAM\EXPLPROD)
This file contains USGS play-specific data for defining dominant resource type, federal fraction and other indicators. This file has been shortened to fit onto one page in the Appendix.

```
USSG Play           GSAM Supply          #  Dev.  Succ. Rate  Res. Expl.   Roy.   Und.
Code                Region                  Curr  n/a   Adv   Type Depth   Rate   Fed.%
0101F               North Alaska         1  80.0  90.0  90.0 1   7577.0   12.5   60.00
0101P               North Alaska         1  80.0  90.0  90.0 1   7577.0   12.5   60.00
0102F               North Alaska         1  80.0  90.0  90.0 1   9300.0   12.5   25.00
0102P               North Alaska         1  80.0  90.0  90.0 1   9300.0   12.5   25.00
0103F               North Alaska         1  80.0  90.0  90.0 1   7076.0   12.5   30.00
0103P               North Alaska         1  80.0  90.0  90.0 1   7076.0   12.5   30.00
0105F               North Alaska         1  80.0  90.0  90.0 1   9300.0   12.5   70.00
0105P               North Alaska         1  80.0  90.0  90.0 1   9300.0   12.5   70.00
0106F               North Alaska         1  80.0  90.0  90.0 1  10000.0   12.5   70.00
0106P               North Alaska         1  80.0  90.0  90.0 1  10000.0   12.5   70.00
0109F               North Alaska         1  80.0  90.0  90.0 1   1573.0   12.5   35.00
0109P               North Alaska         1  80.0  90.0  90.0 1   1573.0   12.5   35.00
0110F               North Alaska         1  80.0  90.0  90.0 1   9300.0   12.5   40.00
0110P               North Alaska         1  80.0  90.0  90.0 1   9300.0   12.5   40.00
0111F               North Alaska         1  80.0  90.0  90.0 1   6515.0   12.5   80.00
0111P               North Alaska         1  80.0  90.0  90.0 1   6515.0   12.5   80.00
0201F               North Alaska         1  80.0  90.0  90.0 1   9300.0   12.5   50.00
0201P               North Alaska         1  80.0  90.0  90.0 1   9300.0   12.5   50.00
0205F               North Alaska         1  80.0  90.0  90.0 1   9300.0   12.5   50.00
0205P               North Alaska         1  80.0  90.0  90.0 1   9300.0   12.5   50.00
0302F               North Alaska         1  80.0  90.0  90.0 1   9300.0   12.5   20.00
0302P               North Alaska         1  80.0  90.0  90.0 1   9300.0   12.5   20.00
0303F               North Alaska         1  80.0  90.0  90.0 1   5402.0   12.5   10.00
0303P               North Alaska         1  80.0  90.0  90.0 1   5402.0   12.5   10.00
0304F               North Alaska         1  80.0  90.0  90.0 2  10980.0   12.5   10.00
0304P               North Alaska         1  80.0  90.0  90.0 2  10980.0   12.5   10.00
```

.
.
.
(continues with other regions)

Data Element	Description	Format
1	5-digit play code	a20
2	GSAM Supply Region	a20
3	State Code	i2
4	Development Success Rate (Current Technology)	f6.1
5	Development Success Rate (Unknown Technology)	f6.1
6	Development Success Rate (Advanced Technology)	f6.1
7	Dominant Resource Type	i2
8	Exploratory Depth (ft.)	f9.1
9	Royalty Rate (%)	f6.1
10	Undiscovered Federal Land Percentage	f8.2
11	H2S Impurity Level (%) (not shown)	f8.4
12	CO2 Impurity Level (%) (not shown)	f8.4
13	N2 Impurity Level (%) (not shown)	f8.4
14	Water Depth (ft.) (not shown)	f8.1
15	Non-associated USGS Reserves (BCF)(not shown)	f13.3

Explanation

NRG database impurity values are used in the Reservoir Performance Module whenever available, otherwise play average values are used to calculate impurity levels. The file also describes the dominant resource type of the play and the non-associated reserves in BCF from USGS.

Intended Uses of PLY_DFN.SPC

The Reservoir performance Module reads the royalty rates and the federal land fractions of undiscovered plays from PLY_DFN.SPC. The impurities' values may be altered to reflect new beliefs, new information, or for sensitivity analysis. Altering the impurity information would affect a play's methane concentration, hence the economics, resulting in a change in the play's reservoirs' MASP. It should be noted that if a NRG data point is not available then all reservoirs in a particular USGS play have the same impurity level as specified in this file.

Table B-5

Input Specification File: REGIONS.DAT (Location: \GSAM\RESVPERF)
This file contains information for the .GSM files to be run through the Reservoir Performance Module and other yes/no switches which open specific files for consistency checks.

```
C*** Number of years
40
C*** Reports to Print
Type Curve Input File (*.tci)            NO
Detailed Pro-forma (*.pro)               NO
Type Curve Output (*.tco)                NO
Reduced Form Proforma (*.prr)            NO
Net Present Value Summary (*.npv)        NO
Print Pressures to *.prd File            NO
Region                          Run Type Curve ??
C*******C************************ C*(t34)
gsam01    c:\gsam\resvperf       YES
gsam02    c:\gsam\resvperf       YES
gsam03    c:\gsam\resvperf       YES
gsam04    c:\gsam\resvperf       YES
gsam05    c:\gsam\resvperf       YES
gsam06    c:\gsam\resvperf       YES
gsam07    c:\gsam\resvperf       YES
```

Counter indicating whether type curves should be run

Directory name for the location of the input/output Reservoir Performance Files

Name of the .GSM file to be run through the reservoir performance model.

In the example above the RP Module will run data for regions 01 through 07 with files gsam01.gsm through gsam07.gsm.

Explanation of REGIONS.DAT

REGIONS.DAT is used to determine which regions/resource types are run through the Reservoir Performance Module. The output options at the top (i.e. reports to print) are primarily for consistency checks. These files will be printed out for each reservoir run through RP, for example 14913002.TCI and 14913002.PRO may be output for GSAMID 02314913002. Because of this, these output files will be very large, and should only be used to test on small groups of reservoirs. The **type curve input** file (.TCI) shows a reservoir's geologic parameters that will be submitted to the type curve routine. The detailed **pro-forma** file (.PRO) shows the projected cash flow over the 40-year period of the model, or the life of the reservoir, whichever is shorter. It calculates these numbers for pay grade 2 only. The **type curve output** (.TCO) shows the projected production rate, open flow potential, bottomhole pressure, wellhead pressure, and water production under primary drilling, re-fracturing, and infill drilling. The **reduced pro-forma** provides a concise economic summary for every reservoir analyzed by the RP Module. The **net present value** file (.NPV) shows the net present value of the cashflow, costs, and revenues of the reservoir as it produces over its lifetime among several cases. This file includes information for permutations of three cases: pay grade (1-3), technology (current or advanced), and drilling (primary, infill, or re-fractured). When the **pressure** to .PRD file option is chosen, the bottomhole and wellhead pressures are printed to the production (.PRD) file. Regardless of this option, the .PRD is always printed, normally without the pressures. The E&P Module does not need the pressures for making investment decisions, so this flag should be set to "NO" if binary database files will be created later.

Intended Uses of REGIONS.DAT

When testing reservoir parameters or RP economics on a small group of reservoirs, activate the debugging files. Use this file to specify the .GSM files to be run.

Table B-6

Input Specification File: RUNSET.DAT (Location: \GSAM\RESVPERF)
This generic file contains run specifications for the RP Module.

```
1
Is this One Line Format
N
Model Start End of Year (e.g. 1997..)
1997
Do history check (N if One line format)
Y
Correction Year for DB Versions (0 = 1997, 1 = 1996, 2 = 1995, etc.)
0
Is this an Env run
N
Env Run for Producing Reservoir
N
Year for Env Run
2020
```

Explanation

The very first line should contain a 1 if the RUNSET.DAT file is to be used. Otherwise, the Module will prompt the user on the screen for the information contained in the file. The undiscovered and Appalachian reservoirs have .GSM files that are in 1-line format, and the discovered .GSM files and Canadian discovered-undeveloped are in 13-line format, so this must be specified by the Y/N toggle.

Because the undiscovered and undeveloped reservoirs have not produced in the past, there is no need for a **history check** of past production. Virtually always a history check will be desired for the discovered reservoirs so that the post-1993 (or whatever start year specified) pressures and production values are consistent with the reservoir's parameters. A history check is done from the latest year of available NRG data (1993 for U.S., 1994 for Canada). But since the model currently begins in 1993, the production history for the Canadian reservoirs must be offset by one year, or the model will consider the reservoir to have been producing for one extra year. For instance, if a reservoir has been calculated to have been producing for 10 years to 1994, in order to get an accurate 1993 starting point for that reservoir, a 1 must be subtracted away so that the reservoir is considered to have been producing for 9 years to 1993. The data for the U.S. discovered reservoirs ends in 1993 so the corresponding RUNSET file must have a 0 for the **model correction year**. The Canadian discovered reservoirs have data that ends in 1994 so that the entry in this case must be a 1. Having this parameter also allows for the incorporation of new, different, or updated data that may have another year as its last.

An **environmental costing routine** may be incorporated in *either* the RP Module or the E&P Module. Environmental compliance costs in the RP Module are set in the COST.DAT file. These costs are region-specific and do not change over time. To run an "environmental" RP run, set the environmental costs in the COST.DAT file for all regions desired. In the "Is this an Env run" line of RUNSET, put a "Y". For environmental compliance costs only to be applied to undiscovered reservoirs, set the "Env Run for Producing Reservoir" line to "N". For undiscovered and discovered reservoirs to be subject to these costs, set the "Env Run for Producing Reservoir" line to "Y" *and* set the "Year for Env Run" line to the desired time for the discovered reservoirs to be effected. For undiscovered reservoirs, environmental costs are applicable from start.

Intended Uses of RUNSET.DAT

RUNSET.DAT should be used to ensure that the RP run is consistent with the desired run features and the data input files, mainly the RP environmental costing. Use environmental costing in the RP Module when sensitivity to environmental cost runs are being performed on regions or groups of reservoirs. These costs do not change over time. The environmental costing in the E&P Module changes over time.

Table B-7

Input Data File: TAX_NAT.DAT (Location: \GSAM\RESVPERF\DATA)
This is a national tax file for the Reservoir Performance Module.

```
C*** U.S. Federal Income Tax Rate
34.0
C*** Canadian Federal Income Tax Rate
28.0
C*** Independent Producer Depletion Rate (%)
100.0
C***  Are Intangible Drilling Costs to be Capitalized? (YES/NO)
YES
C***  Are Other Intangibles to be Capitalized? (YES/NO)
YES
C***  Include environmental Costs? (YES/NO)
YES
C***  Are Environmentals to be Capitalized? (YES/NO)
NO
C***  Implement Alternative Minimum Taxes? (YES/NO)
NO
C***  Allow AMT Taxes Paid to be Used as Credits in Future Years? (YES/NO)
YES
C***  Six Month Amortization Rate (%)
50.0
C***  Intangible Drilling Cost Preference Deduction (%)
100.0
C***  ACE Rate (%)
70.0
C***  Maximum Alternative Minimum Tax Reduction for Independents
0.0
C***  Alternative Minimum Tax RATE (%)
20.0
C***  Expense Environmental Costs? (YES/NO)
NO
C***  Allow Net Income Limitations? (YES/NO)
NO
C***  Net Income Limitation Limit (%)
40.0
C***  Percent Depletion Rate (%)
0.0
C***  Percent of Intan. Inv. to Capitalize (%)
30.0
C***  EOR Tax Credit Rate (%)
15.0
C***  Allow G&G Depletable Tax Credit? (YES/NO)
NO
C***  G&G Depletable Tax Credit Rate (%)
10.0
C***  Allow Tax Credit for Expensed G&G? (YES/NO)
NO
C***  G&G Intangible Tax Credit Rate (%)
15.0
C***  Allow Lease Acq. Depletable Tax Credit? (YES/NO)
NO
C***  Lease Acq. Depletable Tax Credit Rate (%)
10.0
C***  Allow Tax Credit for Expensed Lease Acq. Costs? (YES/NO)
```

```
NO
C***  Tax Credit Rate for Expensed Lease Acq. Costs (%)
15.0
C***  Allow Tangible Development Tax Credit? (YES/NO)
NO
C***  Tangible Development Tax Credit Rate (%)
15.0
C***  Allow Intangible Drilling Cost Tax Credit? (YES/NO)
NO
C***  Intangible Drilling Cost Tax Credit Rate (%)
15.0
C***  Allow Other Intangible Tax Credit? (YES/NO)
NO
C***  Other Intangible Tax Credit Rate (%)
15.0
C***  Allow Environmental Tangible Tax Credit? (YES/NO)
NO
C***  Environmental Tangible Tax Credit Rate (%)
20.0
C***  Allow Environmental Intangible Tax Credit? (YES/NO)
NO
C***  Environmental Intangible Tax Credit Rate (%)
20.0
C***  Allow Environmental Operating Cost Tax Credit? (YES/NO)
NO
C***  Environmental Operating Cost Tax Credit Rate (%)
20.0
C***  Allow Tax Credit On Tangible Investments? (YES/NO)
NO
C***  Number of Years for Tax Credit on Tangible Investments
20
C***  Allow Tax Credit On Intangible Investments? (YES/NO)
NO
C***  Number of Years for Tax Credit on Intangible Investments
15
C***  Percent of G&G Depleted (%)
16.17
C***  Allow  Forgiveness of State Taxes? (YES/NO)
NO
C***  Number of Years for Forgiveness of State Taxes
10
C***  Percent Lease Acquisition Cost Capitalized
100.0
```

Explanation of TAX_NAT.DAT

The **federal income tax rate** has been estimated at 34% for integrated U.S. oil and gas companies, and at 28% for Canadian companies. It may be changed if desired. Many of the features of this file deal with accounting procedures.

Intended Uses of TAX_NAT.DAT

TAX_NAT.DAT can be used for various policy runs by utilizing the parameters in the file. Changing these parameters does not change production rates, pressures, etc., so that *.BIN files can be employed (no type curve) for faster run times, if available. Various tax credits, AMT analysis, and lease acquisition treatments (capitalized vs. expensed) could be performed by changing the parameters in this file.

Table B-8

Input Data File: TAXES.DAT (Location: \GSAM\RESVPERF\DATA)
This file contains tax rates by state/district.

```
C*** State Tax Rates - Oil Severance Rates - Gas Severance Rates
C*** Number of Regions (Excluding Default - 99)
60
Code     State   Oil     Oil    Gas     Gas     Ad-Valorem
C*        (%)    (%)   ($/Bbl)  (%)   ($/MCF)  Tax (% of prod)
0100     0.00  10.00   0.000  10.00  0.00000  0.00        :Code| State Tax| Oil Sev. Tax| Gas Sev. Tax|
0105     5.00  10.00   0.000  10.00  0.00000  0.00        :0100:ALABAMA FED.OFFSHORE,0105:STATE OFFSHORE
0110     5.00  10.00   0.000  10.00  0.00000  0.00        :ALABAMA ONSHORE              :             :
5000     0.00  15.00   0.004  10.00  0.00008  0.00        :ALASKA SOUTH FED. OFFSHORE   :             :
5005     9.40  15.00   0.004  10.00  0.00008  0.00        :ALASKA SOUTH STATE OFFSHORE  :             :
5010     9.40  15.00   0.004  10.00  0.00008  0.00        :ALASKA SOUTH ONSHORE         :             :
5050     9.40  15.00   0.004  10.00  0.00008  0.00        :ALASKA NORTH ONSHORE         :             :
2        9.00   0.00   0.000   0.00  0.00000  0.00        :ARIZONA                      :             :
0310     6.50   5.00   0.000   0.00  0.00300  0.00        :ARKANSAS SOUTH               :             :
0350     6.50   5.00   0.000   0.00  0.00300  0.00        :ARKANSAS NORTH               :             :
36       10.0   5.00   0.000   5.00  0.00000  0.00        :OREGON                       :             :
0400     0.00   0.00   0.025   0.00  0.02500  0.00        :CALIFORNIA FED. OFFSHORE     :             :
0405     9.30   0.00   0.025   0.00  0.02500  0.00        :CALIFORNIA STATE OFFSHORE    :             :
0410     9.30   0.00   0.025   0.00  0.02500  0.00        :CALIFORNIA CENTRAL VALLEY    :             :
0450     9.30   0.00   0.025   0.00  0.02500  0.00        :CALIFORNIA COASTAL           :             :
0490     9.30   0.00   0.025   0.00  0.02500  0.00        :CALIFORNIA LOS ANGELES BASIN :             :
5        5.00   5.17   0.000   5.17  0.00000  0.00        :COLORADO                     :             :
0900     0.00   8.00   0.000   0.00  0.12400  0.00        :FLORIDA FED. OFFSHORE        :             :
0910     5.50   8.00   0.000   0.00  0.12400  0.00        :FLORIDA ONSHORE              :             :
12       4.80   0.00   0.000   0.00  0.00000  0.00        :ILLINOIS                     :             :
13       4.50   1.00   0.000   0.00  0.00000  0.00        :INDIANA                      :             :
15       4.00   8.00   0.000   8.00  0.00000  0.00        :KANSAS                       :             :
1700     0.00  12.50   0.000   0.00  0.07000  0.00        :LOUISIANA FED. OFFSHORE      :             :
1705     8.00  12.50   0.000   0.00  0.07000  0.00        :LOUISIANA STATE OFFSHORE     :             :
1710     8.00  12.50   0.000   0.00  0.07000  0.00        :LOUISIANA SOUTH              :             :
1750     8.00  12.50   0.000   0.00  0.07000  0.00        :LOUISIANA NORTH              :             :
21       2.30   6.60   0.000   5.00  0.00000  0.00        :MICHIGAN                     :             :
2300     0.00   6.00   0.000   6.00  0.00000  0.00        :MISSISSIPPI FED. OFFSHORE    :             :
2310     5.00   6.00   0.000   6.00  0.00000  0.00        :MISSISSIPPI ONSHORE          :             :
25       6.75   5.00   0.000   2.65  0.00000  0.00        :MONTANA                      :             :
26       7.81   3.00   0.000   3.00  0.00000  0.00        :NEBRASKA                     :             :
3010     7.60   7.09   0.000   9.755 0.00000  0.00        :NEW MEXICO SOUTHEAST         :             :
3050     7.60   7.09   0.000   9.755 0.00000  0.00        :NEW MEXICO NORTHWEST         :             :
33      10.50   5.00   0.000   2.00  0.00000  0.00        :NORTH DAKOTA                 :             :
3510     6.00   7.00   0.000   7.00  0.00000  0.00        :OKLAHOMA SOUTHWEST           :             :
3520     6.00   7.00   0.000   7.00  0.00000  0.00        :OKLAHOMA SOUTHEAST           :             :
3530     6.00   7.00   0.000   7.00  0.00000  0.00        :OKLAHOMA NORTHEAST           :             :
3540     6.00   7.00   0.000   7.00  0.00000  0.00        :OKLAHOMA NORTH CENTRAL       :             :
3550     6.00   7.00   0.000   7.00  0.00000  0.00        :OKLAHOMA NORTHWEST           :             :
40       6.00   4.50   0.000   4.50  0.00000  0.00        :SOUTH DAKOTA                 :             :
4200     0.00   4.60   0.000   7.50  0.00000  0.00        :TEXAS FED. OFFSHORE          :             :
4205     0.00   4.60   0.000   7.50  0.00000  0.00        :TEXAS STATE OFFSHORE         :             :
4210     0.00   4.60   0.000   7.50  0.00000  0.00        :TEXAS RRC DISTRICT 1         :             :
4220     0.00   4.60   0.000   7.50  0.00000  0.00        :TEXAS RRC DISTRICT 2         :             :
4230     0.00   4.60   0.000   7.50  0.00000  0.00        :TEXAS RRC DISTRICT 3         :             :
4240     0.00   4.60   0.000   7.50  0.00000  0.00        :TEXAS RRC DISTRICT 4         :             :
4250     0.00   4.60   0.000   7.50  0.00000  0.00        :TEXAS RRC DISTRICT 5         :             :
4260     0.00   4.60   0.000   7.50  0.00000  0.00        :TEXAS RRC DISTRICT 6         :             :
4270     0.00   4.60   0.000   7.50  0.00000  0.00        :TEXAS RRC DISTRICT 7B        :             :
4275     0.00   4.60   0.000   7.50  0.00000  0.00        :TEXAS RRC DISTRCIT 7C        :             :
4280     0.00   4.60   0.000   7.50  0.00000  0.00        :TEXAS RRC DISTRICT 8         :             :
4285     0.00   4.60   0.000   7.50  0.00000  0.00        :TEXAS RRC DISTRICT 8A        :             :
4290     0.00   4.60   0.000   7.50  0.00000  0.00        :TEXAS RRC DISTRICT 9         :             :
4295     0.00   4.60   0.000   7.50  0.00000  0.00        :TEXAS RRC DISTRICT 10        :             :
43       5.00   5.00   0.000   5.00  0.00000  0.00        :UTAH                         :             :
47       9.00   5.00   0.000   5.00  0.00000  0.00        :WEST VIRGINIA                :             :
49       0.00   5.06   0.000   5.06  0.00000  7.662       :WYOMING                      :             :
5300    15.50   4.50   0.000   4.50  0.00000  0.00        :CANADA  (Alberta)            :
5301    16.50   9.00   0.000   9.00  0.00000  0.00        :CANADA  (British Columbia)   :
5302    17.00  10.00   0.000  10.00  0.00000  0.00        :CANADA  (Saskatchewan)        :
9900     10.0   5.00   0.000   5.00  0.00000  0.00        :DEFAULT RATES                :             :
```

Description of File: TAXES.DAT

Data Element	Description	Format
1	4-digit state code	Free format - integer
2	State income tax (%)	Free format - real
3	Oil severance tax (% revenue)	Free format - real
4	Incremental oil severance tax ($/Bbl)	Free format - real
5	Gas severance tax (% revenue)	Free format - real
6	Incremental gas severance tax ($/MCF)	Free format – real
7	Ad-valorem taxes (% of production)	Free format – real
8	Name of the four-digit state code	Free format

Explanation

TAXES.DAT contains state income taxes, oil and gas severance taxes, and ad-valorem taxes. These numbers are taken from state publications (Chamber of Commerce data) for the U.S. and from NEB (National Energy Board) publications for Canadian provinces. The values used are for integrated oil and gas companies. Although the tax structures for independent operators are a little different, the NRG data currently restricts identifying these operators. Once the NRG data for "Dominant Operator Type" for reservoirs becomes populated, the tax treatments for independent operators can change.

Intended Uses of TAXES.DAT

Sensitivity runs on alternative tax credit scenarios could be run by changing the parameters in the table.

Table B-9

Input Data File: TECH.DAT (Location: \GSAM\RESVPERF\DATA)
This is a technology file for the Reservoir Performance Module. It generally contains specifications for two technology cases.

```
C*** Number of Technologies
2
C*** Name of Technology One
Current Technology
C*** Dry Hole Probability (%)
20.0
C*** Year to drill Infill wells for Water Drive Reservoirs
5.0
c**** number of regions for proration
7
c  Region Number and Proration (Fraction)
1   0.07
9   0.10
13  0.25
14  0.25
15  0.25
16  0.25
17  0.25
c   Default proration factor
99  0.10
c****number of states for state specific proration
0
c proration factors by state
c**** Number of different regions for Pay Continuity Enhancement
1
c  Pay Enhancement (Based on BEG study)
1  1.0
c  Default for Pay Enhancement
99  1.0
c**** Number of different regions for System Pressure
2
c  Minimum system pressures by region
1    20.
9    20.
c  Default for Minimum System Pressure
99  150.
c  Number of Reservoir Types to Describe Well Performance Factors
6
c  Vertical Well Skin Factors for Reservoir Types 1 through Number above (For Vertical Well,
Horz Skin calculated in the model)
8.    7.    7.    7.    7.    10.
c  Well Radius for Reservoir Types 1 through Number Above (Assume 9 inch hole)
0.354  0.354  0.354  0.354  0.354  0.354
c  Fracture Half Lengths for Reservoir Types 1 through Number above
0.     300.   300.   300.    0.    150.
c  Fracture Conductivity for Reservoir Types 1 through Number above
0.      100.   100.   100.   0.      50.
c number of regions for horizontal wells
0
c enter horizontal well info
c***** Number of different regions for tubing diameter
1
c ****** Enter tubing size by region (inches) (Assume 2 7/8 tubing)
1   1.4
c ****** Enter tubing size default (inches)
99  1.995
c end of technology
C*** Name of Technology Two
Advanced Technology
```

```
C*** Dry Hole Probability (%)
10.0
C*** Year to drill Infill wells for Water Drive Reservoirs
4.0
c**** number of regions for proration
7
c  Region Number and Proration (Fraction)
1   0.10
9   0.10
13  0.30
14  0.30
15  0.30
16  0.30
17  0.30
c   Default proration factor
99  0.15
c****number of states for state specific proration
0
c state specific proration
c**** Number of different regions for Pay Continuity Enhancement
1
c  Pay Enhancement
1  1.2
c  Default for Pay Enhancement
99  1.2
c**** Number of different regions for System Pressure
2
c  Minimum system pressures by region
1    20.
9    20.
c  Default for Minimum System Pressure
99  100.
c  Number of Reservoir Types to Describe Well Performance Factors
6
c  Vertical Well Skin Factors for Reservoir Types 1 through Number above
 2.     3.     3.     3.     3.    6.
c  Well Radius for Reservoir Types 1 through Number Above (Assume 9 inch hole)
0.354  0.354  0.354  0.354  0.354  0.354
c  Fracture Half Lengths for Reservoir Types 1 through Number above
0.     600.   600.   600.    0.    400.
c  Fracture Conductivity for Reservoir Types 1 through Number above
0.     1000.  1000.  1000.   0.     500.
c number of regions for horizontal wells
0
c enter horizontal well info
c***** Number of different regions for tubing diameter
1
c ****** Enter tubing size by region (inches) (Assume 2 7/8 inches tubing)
1   1.4
c ****** Enter tubing size default (inches)
99  1.995
c end of technology
```

Explanation

The number of technologies (1 or 2) and the names in TECH.DAT should correspond to those in COST.DAT. If the user wants to run only one technology then it should be set in the TECH.DAT file, and does not need to be set in COST.DAT.

The development **dry hole rate** is used in determining the number of dry holes drilled.

The **year to drill infill wells for water drive reservoirs** parameter is in place because the type curve for water drive reservoirs is designed in such a manner that it needs the year (from start of production) when first infill drilling is performed. This value is 5 years, but may be changed. For all other resource types, the type curve computes the year for infill driling, which is the year in which the primary wells can no longer sustain the constant production rate.

Proration rates are defined by region (for Appalachia specified by state) and indicate the performance advantage that horizontal wells have over vertical wells. Generally, it is assumed that horizontal wells produce at a faster rate than vertical wells. It is assumed that on average horizontal wells can produce at 25% to 30% higher rates than vertical wells, therefore the default proration rate is 25% for all vertical wells. This means that if the proration rate is not specified for a region, a proration value of 25% is used. In addition, it is assumed that advanced technology improves the proration anywhere from 10% to 40% depending upon the region.

The **pay continuity enhancement factor** is based on the assumption that under normal conditions, only 80% of the total pay is contacted by drilling at well spacing of 320 acres and greater. As well spacing decreases, the pay contacted increases. A value of 1.0 for the pay continuity enhancement factor (specified in TECH.DAT file) changes the pay continuity according to the following equation (the parameter is in the denominator):

Well Spacing	% Pay Contacted
320 acres + up	100 - (20/1) = 80%
160 acres	100 - (12/1) = 88%
80 acres	100 – (8/1) = 92%
40 acres	100 – (3/1) = 97%

A value of 1.2 (the value under advanced technology) in the pay continuity enhancement factor changes the continuity as follows:

Well Spacing	% Pay Contacted
320 acres + up	100 - (20/1.2) = 83.3%
160 acres	100 - (12/1.2) = 90.0%
80 acres	100 – (8/1.2) = 93.3%
40 acres	100 – (3/1.2) = 97.5%

Note that no pay grade improvement is allowed for U.S. coalbed methane reservoirs in advanced technology.

The region-specific **minimum system pressure** is normally only specified for Appalachia. It is assumed that for the Appalachia region gas reservoirs can be produced when the system pressure reaches very close to atmospheric pressure. This information was obtained from various gas operators in the Appalachian basin. For all other regions the minimum pressure that could be sustained at the surface is 250 psia.

The **skin factors** are specified by resource type (1 to 6). (Note 1=conventional, 2=tight with hydraulic fracture, 3=naturally fractured, 4=naturally fractured with hydraulic fracture, 5=water drive, 6=coalbed). This includes the skin related to non-darcy flow, completion technology, and other factors that increase the pressure drop around the wellbore. For tight reservoirs (2), use of fracture reduces the skin to 7, in this case. These skin factors are for vertical wells. For analyzing horizontal well behavior, the vertical well skin factors are used to calculate equivalent horizontal well skin based on vertical permeability, horizontal permeability, net pay, and horizontal well length.

A 9 inch (outside diameter) well that relates to a 4.248 inch (0.354 feet) inside radius is assumed in all cases.

Current practices create on average 300 ft **fractures** (half lengths). The finite **conductivity fractures** are assumed to have a conductivity value of around 100 md-ft. For coalbed methane, the value is lower.

The horizontal technology file lists the **horizontal well length**, which in this case is assumed to be 1000 feet. The first column shows GSAM region name. The second shows whether horizontal wells are an option or not (1 being horizontal well application, 0 meaning vertical wells). The third column indicates the length of the horizontal section of the wellbore. The vertical wells situated in Appalachia are assumed to be draining from a 2 7/8 inch (diameter) tubing, which means that there is 1.4 inches of tubing radius in Appalachia. For all other regions a 4 inch diameter tubing (1.995 inches inside radius) is used for vertical wells.

Intended Use of TECH.DAT

The parameters in TECH.DAT can be changed to reflect assumptions about reservoir technology. The numbers may be formatted in any fashion (decimal, placement, etc.) but the text must remain at the same location.

Table B-10

OUTPUT File: *.SUM

This file is a summary file for current technology output for each .GSM input file (also, *.ASM contains the same information for advanced technology). It has been shortened to fit to one page. GSAM02.SUM is below.

```
Primary Wells Only (Current Tech.)
 GSAMID        Pay    Res.      OGIP    # Wells   MASP      NPV DRL    NPV Tax       Tax Diff
02314901P001 C    1     .000        .000        0.    99.00       .000       .000       .000
02314901P001 C    2    3.155       6.331        3.    99.00      4.390     -1.431      -.427
02314901P001 C    3     .732       1.470        1.    99.00      1.463      -.511      -.192
02314901P002 C    1     .000        .000        0.    99.00       .000       .000       .000
02314901P002 C    2    9.271      12.721        2.      .81      3.203      3.402      6.560
02314901P002 C    3     .000        .000        0.    99.00       .000       .000       .000
02314901P003 C    1     .000        .000        0.    99.00       .000       .000       .000
02314901P003 C    2    8.269      50.561        1.     1.64      2.134       .590      1.774
02314901P003 C    3     .000        .000        0.    99.00       .000       .000       .000
02314901P004 C    1     .000        .000        0.    99.00       .000       .000       .000
02314901P004 C    2   26.026      47.373        3.      .67      5.674      7.753     14.590
02314901P004 C    3     .000        .000        0.    99.00       .000       .000       .000
02314901P005 C    1    7.737      21.878        5.    99.00      2.816     -1.039      -.193
02314901P005 C    2   14.519      41.082       14.    99.00      7.884     -3.193      -.969
02314901P005 C    3    4.805      13.613        7.    99.00      3.942     -1.695      -.632
02314901P006 C    1     .000        .000        0.    99.00       .000       .000       .000
02314901P006 C    2    5.232     101.576        1.    99.00       .000       .000       .000
02314901P006 C    3     .000        .000        0.    99.00       .000       .000       .000
02314901P007 C    1   11.808      26.983        3.     6.69      1.982      -.350       .158
02314901P007 C    2   20.617      47.121        8.    99.00      5.286     -1.249      -.055
02314901P007 C    3    6.993      15.991        4.    99.00      2.643      -.723      -.175
02314901P008 C    1     .000        .000        0.    99.00       .000       .000       .000
02314901P008 C    2   36.872     130.366        4.     3.95      3.367      -.448       .880
02314901P008 C    3    6.058      21.458        1.     6.00       .842      -.206       .080
02314901P009 C    1     .000        .000        0.    99.00       .000       .000       .000
02314901P009 C    2   10.964      23.449        3.    99.00       .000       .000       .000
02314901P009 C    3    2.372       5.070        1.    99.00       .000       .000       .000
02314901P010 C    1     .000        .000        0.    99.00       .000       .000       .000
02314901P010 C    2    6.160      87.800        1.    99.00       .000       .000       .000
02314901P010 C    3     .000        .000        0.    99.00       .000       .000       .000
```

Description of File: *.SUM

Data Element	Description
1	GSAMID (For undiscovered resource it is FSC in a play, for discovered producing it is the reservoir itself)
2	Current (“C”) technology type
3	Pay grade (Pay)
4	Technically recoverable reserves by pay grade (Res.)
5	Original gas in place by pay grade (OGIP)
6	# of wells that could be drilled OR should be available based on acreage and well spacing (# Wells)
7	Minimum acceptable supply price (MASP)
8	NPV of drilling costs, no exploration costs ($MM)
9	NPV of total taxes paid (fed., state, severance) ($MM)
10	Difference in NPV of taxes when calculated at $5/Mcf and $2/Mcf ($MM)

Table B-11

OUTPUT File: *.CUR

This file is a summary output file for current technology for each .GSM input file (also, *.ADV contains the same information for advanced technology). GSAM02.CUR is shown below.

```
Summary of Primary Wells Only For Current Tech.
 GSAMID       STATE   DEPTH   MASP     Model    Reported        Teh.Rec.    Reported   Model Reported     Model       Model  Total Proved    CG9R97     RepOGIP       MOGIP
                              PG 2 1997 Prod.  1997 Prod.      Prim. Res.    Tot.Res.   Wells   Wells   Spacing   Est. Area          Area
02314901P001    2310   13415.  99.00     .10099       .1010          .7070       .1890       4       0      160.00        566.          566.     6.153       7.890       7.802
02314901P002    2310   14012.    .81    1.14661       .0010         9.2713       .0030       2       0       80.00        160.          560.      .867      12.809      12.721
02314901P003    2310   15949.   1.64     .21700       .2170         4.5569       .7570       1       0      640.00        640.         1280.     3.929      20.126      50.561
02314901P004    2310   15126.    .67    5.77300      5.7730        16.4417     29.4110       3       0      640.00       1783.         1783.     9.589      47.797      47.373
02314901P005    4260    7205.  99.00     .20399       .2040         2.6519      1.6670      26       0      320.00       8400.        16000.    24.613      76.231      76.573
02314901P006    4260   10525.  99.00     .00000       .1950          .0000      1.3680       0       0      320.00        320.        16000.     5.232      22.781     101.576
02314901P007    4260    8244.  99.00     .48300       .4830         1.9320      3.7770      15       0      320.00       4808.        10000.    68.613      90.264      90.096
02314901P008    4260    9816.   3.95     .22400       .2240         3.3599      2.2360       5       0      640.00       3019.        10000.    39.794     151.905     151.823
02314901P009    4260    8448.  99.00     .00000       .0220          .0000       .2460       0       0      640.00       2400.        10000.    13.644      29.865      28.519
02314901P010    4260    8863.  99.00     .00000       .0190          .0000       .0630       0       0      640.00        640.        10000.     6.160      21.710      87.800
02314901P011    4260    8925.   3.31     .11099       .1110         1.2210       .6570       2       0      640.00       1433.         6000.     7.683      28.451      27.291
02314905P001     110   15340.   3.51    1.88100      1.8810        24.4530     12.8710      16       0      640.00      10240.        10240.    89.129     388.797     482.721
02314910P001     110   15122.    .83   11.84500     11.8450       238.6807     94.7320      27       0      640.00      17280.        17280.   292.268    1496.366    1899.190
02314910P002     110   18061.    .48   12.15400     12.1540       453.1140    135.1310      13       0      640.00       8320.         8320.   164.869    1581.717    1628.067
02314910P003     110   15090.   4.82     .71600       .7160         3.8702      5.3270       5       0      320.00       1703.         1703.     2.923      13.308      14.941
02314912P001     110   16028.    .27    5.93400      5.9340       126.0622     24.8180       4       0      640.00       2417.         3200.    23.182     307.089     335.702
02314912P002    2310   17405.    .90    1.06600      1.0660         5.3300      8.1700       1       0      640.00        640.         1920.    30.782     182.685     215.363
02314912P003     110   16016.   9.79     .38399       .3840          .8102      1.7600       2       0      640.00       1153.         1153.     7.540      20.024      24.394
02314912P004     110   16407.   1.96     .34000       .3400         2.3800      1.0220       1       0      640.00        640.         1280.     4.558      32.061      35.662
02314912P005     110   16616.   3.61     .10100       .1010         3.9390       .2030       1       0      640.00        640.         1280.     4.477      26.197      53.097
02314912P006    1750    9777.    .42    3.80200      3.8020        68.4359     26.4760       6       0      160.00        905.        30000.    96.498     448.134     457.128
02314912P007     310    9116.   4.82    1.10700      1.1070        43.1730      9.3960      42       0      160.00       6720.        10000.   589.019    1350.030    1397.851
02314912P008    1750   10826.  99.00     .00000      2.0990          .0000     15.7590       0       0      320.00       1011.        25000.    23.993     138.934     141.071
02314912P009     310    7357.  99.00     .00000       .0320          .0000       .1180       0       0      160.00        960.          960.     6.662      17.925      15.806
02314912P010     310    7774.  99.00     .12200       .1220         4.7580       .8620      21       0       40.00        840.          840.     6.698      69.121      67.206
02314912P011    1750   10211.  99.00     .04000       .0400          .6400       .2690       3       0      160.00        479.         3360.     3.511      13.880      14.019
02314912P012    1750    9948.   1.96     .64000       .6400         5.7600      4.5840       5       0      160.00        821.         7000.    14.700      66.700      67.072
02314912P013    1750   10369.   9.50     .20100       .2010         2.2110       .5910       9       0      320.00       2800.         3400.    16.115      50.716      52.880
02314912P014    1750   10370.  99.00     .00000       .0260          .0000       .0820       0       0      320.00        949.         1920.     5.138      18.450      18.603
02314912P015    1750   10316.  99.00     .00000       .0940          .0000       .4470       0       0      640.00        640.         5760.     6.122      24.503      29.078
02314912P016     310   11026.   4.13     .19101       .1910         4.9660      1.0670       5       0      640.00       3027.         3200.    32.517     124.436     122.575
02314912P017    1750   11828.  99.00     .00000       .0270          .0000       .1030       0       0      640.00       3073.         8000.    23.908      80.699      82.772
02314912P018    1750   11497.  99.00     .00000       .0800          .0000       .4630       0       0      320.00        660.         6400.     4.397      17.220      17.622
02314912P019     310   10833.  99.00     .00000      9.4370          .0000     42.1670       0       0      320.00      10841.        15040.   158.833     710.145     732.846
02314912P020     310   11188.  99.00     .00000       .0580          .0000       .1110       0       0      320.00       1920.         1920.    39.309      86.236      89.903
02314912P021     310    9973.  99.00     .31900       .3190        12.4409      2.0870      30       0      160.00       4800.         4800.    41.149     129.321     126.574
02314912P022    1750   10423.  99.00     .00000       .9170          .0000      5.4920       0       0      320.00       2420.         6000.    29.878     129.054     130.825
02314912P023    1750   11645.  99.00     .00000       .2390          .0000      1.3270       0       0      320.00        320.         6000.     4.222      19.958     197.079
02314912P024    1750   10198.  99.00    1.16600      1.1660        13.8932      7.5920       8       0      160.00       4640.         4640.    12.358     147.682     151.422
02314912P025     310   11002.   8.57     .10400       .1040          .9360       .3840       4       0      640.00       2560.         2560.    10.488      39.381      38.856
02314912P026     310   11099.  99.00     .71657      1.9970          .8575      9.9900       5       0      320.00       1600.         1600.     8.550      16.245      16.597
02314912P027    4260   12469.  99.00     .00000      1.0250          .0000      9.2320       0       0      320.00       2176.         9500.    48.368     221.165     229.023
02314912P028    4250   12672.  99.00     .00000       .6620          .0000      3.1620       0       0      640.00        640.         6000.      .678      10.892      11.413
```

Data Element	Description
1	GSAMID
2	State in which GSAM ID is located
3	Depth of formation (feet)
4	Minimum acceptable supply price (MASP) $/Mcf, pay grade 2, development drilling
5	RP calculated 1997 production (Bcf)
6	Reported NRG 1997 production (Bcf)
7	RP calculated technically recoverable primary reserves (Bcf)
8	NRG Reported reserves as of 1997
9	Wells needed to be drilled in reservoir based on well spacing and acreage available—for discovered producing reservoirs, # of wells that should be available
10	# of wells currently operating (from NRG database)
11	Well spacing (acres) at which development drilling should occur
12	RP calculated area (acres)
13	Area from NRG database
14	Cumulative gas produced to 1997 (Bcf)
15	OGIP from NRG database (Bcf)
16	RP calculated OGIP (Bcf_

Table B-12

OUTPUT File: *.PRD
This file is the output file that shows production and operating costs for each reservoir in .GSM input file. It has been shortened to fit to one page.

GSAM ID **EIA Code** **Technology Type (C: current, A: advanced)**

Pay Grade (1-3)

O&M stream by year ($/Mcf)

```
02314901P001   2215     C 1 P O&M   1     .0000
02314901P001   2215     C 1 P GASP  1     .0000
02314901P001   2215     C 2 P O&M   7     .1144     .1141     .1141     .1141     .1141     .1141
02314901P001   2215     C 2 P GASP  7     .0829     .0821     .0820     .0820     .0820     .0820
02314901P001   2215     C 3 P O&M   7     .0347     .0350     .0350     .0350     .0350     .0350
02314901P001   2215     C 3 P GASP  7     .0181     .0189     .0190     .0190     .0190     .0190
02314901P001   2215     C 1 R O&M   1     .0000
02314901P001   2215     C 1 R GASP  1     .0000
02314901P001   2215     C 2 R O&M   7     .1144     .1141     .1141     .1141     .1141     .1141
02314901P001   2215     C 2 R GASP  7     .0829     .0821     .0820     .0820     .0820     .0820
02314901P001   2215     C 3 R O&M   7     .0347     .0350     .0350     .0350     .0350     .0350
02314901P001   2215     C 3 R GASP  7     .0181     .0189     .0190     .0190     .0190     .0190
02314901P001   2215     C 1 I O&M   1     .0000
02314901P001   2215     C 1 I GASP  1     .0000
02314901P001   2215     C 2 I O&M   6     .1144     .1141     .1141     .1141     .1141     .1141
02314901P001   2215     C 2 I GASP  6     .0828     .0821     .0820     .0820     .0820     .0820
02314901P001   2215     C 3 I O&M   6     .0347     .0350     .0350     .0350     .0350     .0350
02314901P001   2215     C 3 I GASP  6     .0181     .0189     .0190     .0190     .0190     .0190
02314901P001   2215     A 1 P O&M   1     .0000
02314901P001   2215     A 1 P GASP  1     .0000
02314901P001   2215     A 2 P O&M   1     .0676
02314901P001   2215     A 2 P GASP  1     .0000
02314901P001   2215     A 3 P O&M   1     .0225
02314901P001   2215     A 3 P GASP  1     .0000
02314901P001   2215     A 1 R O&M   1     .0000
02314901P001   2215     A 1 R GASP  1     .0000
02314901P001   2215     A 2 R O&M   1     .0676
02314901P001   2215     A 2 R GASP  1     .0000
02314901P001   2215     A 3 R O&M   1     .0225
02314901P001   2215     A 3 R GASP  1     .0000
02314901P001   2215     A 1 I O&M   1     .0000
02314901P001   2215     A 1 I GASP  1     .0000
02314901P001   2215     A 2 I O&M  10     .0971     .1646     .1646     .1646     .1646     .1646
02314901P001   2215     A 2 I GASP 10     .0824     .0820     .0820     .0820     .0820     .0820
02314901P001   2215     A 3 I O&M  10     .0292     .0519     .0519     .0519     .0519     .0519
02314901P001   2215     A 3 I GASP 10     .0186     .0190     .0190     .0190     .0190     .0190
02314901P002   2215     C 1 P O&M   1     .0000
```

Number of years for production without economics considerations

Type (O&M: Operating and Maintenance Cost ($/Mcf), GASP: Gas production(Bcf/yr.))

Case (P: Primary, R: Re-fracture, I: Infill)

Table B-13

OUTPUT File: *.DEC This file has reservoir decisions, including summary economics, one for each .GSM input file.

Example of a Typical Reservoir Decision File

Reservoir 1

GSAM ID	EIACODE	CASE	Resv.	OGIP	# Wells	MASP	Tot. Cap.	NPV Prod.	NPV Exp.	NPV Inv.	NPV Drill	NPV Non-Drl	NPV Tax	Chg. Exp.	Chg. Inv.	Chg. Tax	Drill Slope	Non-Drl Slope	Feet	H20 Depth	Window Year	Productive Life
06114702P014	0	C 1 P *	71	130.7	2	1	17.4	44.446	26.954	19.129	18.782	0.347	19.124	21.489	10.298	40.659	0.696	0.387	19500	0	6	31
06114702P014	0	C 1 R	71	130.7	2	1	17.4	44.662	27.219	19.129	18.782	0.347	19.199	21.676	10.298	40.843	0.693	0.391	19500	0	6	29
06114702P014	0	C 1 I	88.8	130.7	4	1.24	34.5	51.455	32.125	30.791	30.444	0.347	19.207	25.228	16.596	45.52	0.563	0.352	19500	0	6	29
06114702P014	0	C 2 P	128.9	237.2	5	1.24	43.5	79.042	48.846	47.798	46.956	0.842	29.532	38.539	25.735	69.929	0.696	0.364	19500	0	6	39
06114702P014	0	C 2 R *	128.9	237.2	5	1.24	43.5	79.884	49.697	47.798	46.956	0.842	29.889	39.154	25.735	70.689	0.693	0.373	19500	0	6	35
06114702P014	0	C 2 I	161.1	237.2	10	1.57	86.1	92.189	59.074	76.954	76.111	0.842	27.302	45.765	41.479	77.775	0.564	0.333	19500	0	6	34
06114702P014	0	C 3 P	46.9	87.1	3	1.95	26.1	25.798	16.831	28.685	28.173	0.512	5.38	12.892	15.444	20.566	0.696	0.317	19500	0	6	40
06114702P014	0	C 3 R *	47.2	87.1	3	1.92	26.1	26.411	17.425	28.685	28.173	0.512	5.648	13.318	15.444	21.126	0.693	0.322	19500	0	6	40
06114702P014	0	C 3 I	59.1	87.1	6	2.49	51.7	30.885	21.273	46.179	45.667	0.512	2.471	15.879	24.89	22.511	0.564	0.282	19500	0	6	40

GSAM ID	EIACODE	CASE	Resv.	OGIP	# Wells	MASP	Tot. Cap.	NPV Prod.	NPV Exp.	NPV Inv.	NPV Drill	NPV Non-Drl	NPV Tax	Chg. Exp.	Chg. Inv.	Chg. Tax	Drill Slope	Non-Drl Slope	Feet	H20 Depth	Window Year	Productive Life
06114702P014	0	A 1 P	76.5	130.7	2	0.85	15.3	51.785	30.696	16.866	16.385	0.481	24.167	24.79	9.064	48.354	0.696	0.429	19500	0	3	28
06114702P014	0	A 1 R *	76.5	130.7	2	0.85	15.3	52.266	31.097	16.866	16.385	0.481	24.399	25.093	9.064	48.805	0.694	0.435	19500	0	3	26
06114702P014	0	A 1 I	92.8	130.7	4	1.06	30.2	63.889	38.679	30.407	29.926	0.481	26.631	30.892	16.376	57.962	0.635	0.435	19500	0	3	23
06114702P014	0	A 2 P	138.8	237.2	5	1	38.3	96.841	57.861	42.13	40.963	1.167	41.838	46.525	22.645	88.631	0.696	0.433	19500	0	3	32
06114702P014	0	A 2 R *	138.8	237.2	5	0.99	38.3	98.819	59.296	42.13	40.963	1.167	42.867	47.644	22.645	90.525	0.695	0.448	19500	0	3	27
06114702P014	0	A 2 I	168.4	237.2	10	1.29	75.5	118.005	72.454	75.983	74.816	1.167	42.847	57.442	40.926	103.665	0.636	0.432	19500	0	3	26
06114702P014	0	A 3 P	50.9	87.1	3	1.41	23	34.309	21.075	25.287	24.578	0.71	11.496	16.684	13.591	29.633	0.696	0.394	19500	0	3	40
06114702P014	0	A 3 R *	51	87.1	3	1.37	23	35.868	22.167	25.287	24.578	0.71	12.32	17.536	13.591	31.136	0.695	0.424	19500	0	3	33
06114702P014	0	A 3 I	61.8	87.1	6	1.88	45.3	42.319	27.101	45.599	44.889	0.71	9.436	21.009	24.56	34.019	0.636	0.391	19500	0	3	33

— — — — —

Reservoir 2

GSAM ID	EIACODE	CASE	Resv.	OGIP	# Wells	MASP	Tot. Cap.	NPV Prod.	NPV Exp.	NPV Inv.	NPV Drill	NPV Non-Drl	NPV Tax	Chg. Exp.	Chg. Inv.	Chg. Tax	Drill Slope	Non-Drl Slope	Feet	H20 Depth	Window Year	Productive Life
06114702P015	0	C 1 P *	148	272.4	3	0.88	26.3	87.048	52.919	28.945	28.254	0.691	39.992	42.094	15.568	81.036	0.696	0.365	19500	0	5	36
06114702P015	0	C 1 R	148.1	272.4	3	0.88	26.3	87.516	53.434	28.945	28.254	0.691	40.175	42.459	15.568	81.449	0.693	0.369	19500	0	5	34
06114702P015	0	C 1 I	185	272.4	6	1.07	52	103.468	64.485	48.243	47.552	0.691	42.802	50.631	25.989	93.827	0.585	0.345	19500	0	5	31
06114702P015	0	C 2 P	258.8	476.4	8	1.09	70.1	158.443	97.561	77.151	75.344	1.806	64.975	77.077	41.499	143.324	0.696	0.369	19500	0	5	40
06114702P015	0	C 2 R *	258.9	476.4	8	1.08	70.1	160.308	99.315	77.151	75.344	1.806	65.809	78.351	41.499	145.036	0.694	0.377	19500	0	5	37
06114702P015	0	C 2 I	323.6	476.4	16	1.37	138.6	187.181	119.078	128.612	126.806	1.806	64.103	92.53	69.288	162.617	0.585	0.343	19500	0	5	35
06114702P015	0	C 3 P	86.9	161.4	4	1.51	35.1	48.759	31.031	38.586	37.672	0.914	15.166	24.075	20.754	41.548	0.696	0.333	19500	0	5	40
06114702P015	0	C 3 R *	87.4	161.4	4	1.49	35.1	50.005	32.117	38.586	37.672	0.914	15.752	24.869	20.754	42.711	0.693	0.34	19500	0	5	40
06114702P015	0	C 3 I	109.4	161.4	8	1.93	69.3	59.048	39.297	64.317	63.403	0.914	12.45	29.83	34.649	47.173	0.585	0.304	19500	0	5	40

GSAM ID	EIACODE	CASE	Resv.	OGIP	# Wells	MASP	Tot. Cap.	NPV Prod.	NPV Exp.	NPV Inv.	NPV Drill	NPV Non-Drl	NPV Tax	Chg. Exp.	Chg. Inv.	Chg. Tax	Drill Slope	Non-Drl Slope	Feet	H20 Depth	Window Year	Productive Life
06114702P015	0	A 1 P	159.5	272.4	3	0.75	23.3	98.132	57.407	25.585	24.648	0.937	47.987	46.731	13.732	92.769	0.696	0.389	19500	0	3	34
06114702P015	0	A 1 R *	159.5	272.4	3	0.75	23.3	98.933	58.045	25.585	24.648	0.937	48.385	47.219	13.732	93.525	0.695	0.395	19500	0	3	31
06114702P015	0	A 1 I	193.4	272.4	6	0.9	45.7	126.27	75.052	45.955	45.018	0.937	57.389	60.58	24.732	117.054	0.635	0.412	19500	0	3	27
06114702P015	0	A 2 P	278.8	476.4	8	0.89	62	187.53	110.166	68.177	65.728	2.448	85.724	89.492	36.595	174.059	0.696	0.419	19500	0	3	34
06114702P015	0	A 2 R *	278.9	476.4	8	0.88	62	191.015	112.608	68.177	65.728	2.448	87.565	91.423	36.595	177.411	0.695	0.431	19500	0	3	30
06114702P015	0	A 2 I	338.2	476.4	16	1.11	121.7	234.855	141.079	122.497	120.049	2.448	95.04	113.268	65.928	211.424	0.636	0.43	19500	0	3	27
06114702P015	0	A 3 P	94.3	161.4	4	1.14	31	62.355	37.214	34.103	32.864	1.239	24.854	29.963	18.304	55.935	0.696	0.397	19500	0	3	40
06114702P015	0	A 3 R *	94.4	161.4	4	1.12	31	64.903	38.901	34.103	32.864	1.239	26.233	31.309	18.304	58.407	0.695	0.418	19500	0	3	36
06114702P015	0	A 3 I	114.5	161.4	8	1.47	60.9	78.514	48.376	61.264	60.024	1.239	25.22	38.311	32.971	67.192	0.636	0.402	19500	0	3	34

— — — — —

Table B-14

Auxiliary OUTPUT File: 1493002.TCI

```
================== GSAM INPUT DATA FILE ===============================
 CASE DESCRIPTION (2 Lines):
GSAM Code: 02314913P002 Technology: Current Technology

 =======================================================================
                      IMPURITIES CONCENTRATIONS                SPEED UP
    GAS      TEMP.   ===========================   TUBING ID    CASES
   GRAVITY   DEG. F      H2S       CO2       CN2      INCHES    1=Y,0=N

   _______   ______   _______   _______   _______   _________   ________
    .8530     385.        .09       .11       .02     1.995        1
 =======================================================================
                    BASIC RESERVOIR INFORMATION
                    ===========================
  PAY    INITIAL  HORIZ  VERT    TOTAL  INIT WATER   NET PAY    WATER
 GRADE  PRESSURE  PERM.  PERM.  POROSITY SATURATION  THICKNESS  SALINITY
  NO.     PSIA     MD     MD    DECIMAL   DECIMAL      FEET       PPM

 _____  ________  _____  _____  ________ __________  _________  ________
   1     21800.   7.60    2.28      .11       .22      150.00    30000.
   2     21800.   3.80    1.14      .11       .22       95.90    30000.
   3     21800.   1.90     .57      .11       .22       66.67    30000.
 =======================================================================
       FRACTURED RESERVOIR INFORMATION
       ===============================
  PAY    MATRIX    MATRIX   NAT'L FRAC
 GRADE    PERM.   POROSITY   SPACING
  NO.      MD      DECIMAL    FEET

 _____  ________  ________  __________
   1       7.60     .1110       .00
   2       3.80     .1110       .00
   3       1.90     .1110       .00
 =======================================================================
                     FIELD DEVELOPMENT INFORMATION
                     =============================
                             WELL TYPE (MODULE NO.)  WELLBORE RADIUS, FT
  PAY                INITIAL _____________________  ____________________
 GRADE DEPTH  AREA   SPACING INITIAL FIRST  SECOND   INIT  FIRST  SECOND
  NO.   FEET  ACRES   ACRES   WELL   INFILL INFILL   WELL  INFILL INFILL

 _____ ______ ______ _______ _______ ______ ______  ______ ______ ______
   1   21998.     160. 160.      1      1      1       .35    .35    .35
   2   21998.     480. 160.      1      1      1       .35    .35    .35
   3   21998.     160. 160.      1      1      1       .35    .35    .35
 =======================================================================
                   FRACTURED AND HORIZONTAL WELL DATA
                   ==================================
       TYPE(VERT=0, HORIZ=1) FRAC Xf/HORIZ LENGTH     FRAC COND, MD_FT
  PAY  _____________________ _____________________   ___________________
 GRADE INIT   FIRST  SECOND    INIT  FIRST  SECOND    INIT  FIRST  SECOND
  NO.  WELL   INFILL INFILL    WELL  INFILL INFILL    WELL  INFILL INFILL

 _____ _____  ______ ______    _____ ______ ______    _____ ______ ______
   1     0      0      0         0.     0.     0.        0.     0.     0.
   2     0      0      0         0.     0.     0.        0.     0.     0.
   3     0      0      0         0.     0.     0.        0.     0.     0.
 =======================================================================
              WATER DRIVE AND UNCONVENTIONAL RESERVOIR DATA
              =============================================
       AQUIFER       MAX
       Re/Rw  TRAP  WATER    0=DRY COAL LOCATION
 PAY   0: 2.5 GAS  >1 BPD/WL 1=WET COAL 0=APPAL.  GAS      LANG    DESOR
GRADE  1:  5  SAT  0-1%INFLX 2=DRY SH.  1=ALA.  CONTENT    PRES    TIME    DENS
 NO.   2: INF DEC. <0 BPM    3=WET SH.  2=WEST. (SCF/T)   (PSIA)   (DAYS) (G/CC
_____  ______ ____ __________ __________ _______ ______       ____     ____    ____
   1   0      .20      -.1      0          0      0.         0.      50.      .00
   2   0      .20      -.1      0          0      0.         0.      50.      .00
   3   0      .20      -.1      0          0      0.         0.      50.      .00
 ========================================================================
                                           WELL CONTROL INFORMATION
                                           ========================
          MAXIMUM     INFILL                SKIN FACTORS      SKIN FOR
         RATE MCFD   DATE FOR     PAY    ___________________ AUTO-REFRAC
 MIN WHP  OR %AOF IF WTR DRIVE    GRADE   INIT  FIRST  SECOND   INITIAL
  PSIA    1 OR LESS  RESERVOIRS    NO.    WELL  INFILL INFILL    WELL

 _______  _________  __________  _____  _____ ______ ______ ___________
    500.     6169.21       .0      1      10.0   9.0    9.0        7.0
                                   2      10.0   9.0    9.0        7.0
                                   3      10.0   9.0    9.0        7.0
 ========================================================================
 TIME STEP CONTROL (MAXIMUM 199 STEPS)
 =====================================
 TIME STEP     CHANGE TIME
   YEARS          YEARS

 _________     ___________
    1.0           40.0
 ========================================================================
```

Table B-15

Auxiliary OUTPUT File: 1493002.PRO

```
Detailed Financial Report
                             GSAM ID: 02314913P002 Tech.: Current Technology   Case: Primary P.G.:  2

                                       Year          1         2         3         4         5         6
                                              ========  ========  ========  ========  ========  ========
Oil Production (MMBO)                             .000      .000      .000      .000      .000      .000
Gas Production (BCF)                              .892      .953      .982      .996     1.003     1.007
Gross Revenues (MM$)                              1.78      1.91      1.96      1.99      2.01      2.01
 Gravity/Trans. Cost Adj.                          .00       .00       .00       .00       .00       .00
Adjusted Revenues                                 1.78      1.91      1.96      1.99      2.01      2.01
 Royalties                                         .22       .24       .25       .25       .25       .25
Net Sales                                         1.56      1.67      1.72      1.74      1.76      1.76
Total Operating Cost                              2.04       .22       .22       .22       .22       .22
Operating Cost/Mcf                                2.29       .23       .22       .22       .22       .22
 G&A on Expensed Items                             .04       .04       .04       .04       .04       .04
 G&A on Capitalized Items                         1.45       .00       .00       .00       .00       .00
 Pressure Maint./Cycling                           .00       .00       .00       .00       .00       .00
 General O&M                                       .17       .17       .18       .18       .18       .18
 Environmental O&M Costs                           .00       .00       .00       .00       .00       .00
 Stimulation Costs                                 .38       .00       .00       .00       .00       .00
 Recompletion Costs                                .00       .00       .00       .00       .00       .00
Intangible Investment                             7.95       .00       .00       .00       .00       .00
 Intang. Exploratory Costs                         .00       .00       .00       .00       .00       .00
 Intang. Development Costs                        7.95       .00       .00       .00       .00       .00
 Other Intangible Costs                            .00       .00       .00       .00       .00       .00
 Environmental Intangible Capital Costs            .00       .00       .00       .00       .00       .00
Portion of Intangibles to Capitalize              2.39       .00       .00       .00       .00       .00
TOTAL INVESTMENTS                                14.50       .00       .00       .00       .00       .00
Tangible Investments                              6.55       .00       .00       .00       .00       .00
 Tang. Exploratory Cost                            .00       .00       .00       .00       .00       .00
 Tang. Development Cost                           5.30       .00       .00       .00       .00       .00
 Environmental                                     .01       .00       .00       .00       .00       .00
 Other Tangible Capital                           1.23       .00       .00       .00       .00       .00
Depreciable/Capitalized Investments               8.94       .00       .00       .00       .00       .00
 Adj. for Federal Tax Credits                      .00       .00       .00       .00       .00       .00
Depreciable/Capitalize Base                       8.94       .00       .00       .00       .00       .00
Depreciation                                      1.27      2.19      1.56      1.12       .80       .80
Depletable G&G/Lease Costs                         .00       .00       .00       .00       .00       .00
 Lease Acq. Cost                                   .00       .00       .00       .00       .00       .00
 G&G Costs                                         .00       .00       .00       .00       .00       .00
 Adjustments for Federal Tax Credits               .00       .00       .00       .00       .00       .00
Depletion Base                                     .00       .00       .00       .00       .00       .00
Expensed G&G/Lease Costs                           .00       .00       .00       .00       .00       .00
 Lease Purchase Cost                               .00       .00       .00       .00       .00       .00
 G&G Costs                                         .00       .00       .00       .00       .00       .00
Net Revenues                                      1.56      1.67      1.72      1.74      1.76      1.76
 Operator Severance Taxes                          .09       .10       .10       .10       .11       .11
 Operating Costs                                  2.04       .22       .22       .22       .22       .22
 Expensed Int.,G&G, and Lease Acq.                5.57       .00       .00       .00       .00       .00
 Depreciation                                     1.27      2.19      1.56      1.12       .80       .80
 Depletion Allowance                               .00       .00       .00       .00       .00       .00
Taxable Income                                   -7.41      -.84      -.17       .30       .63       .64
 Tax Credit Addback                                .00       .00       .00       .00       .00       .00
 Intangible Addback                                .00       .00       .00       .00       .00       .00
 G&G/Lease Addback                                 .00       .00       .00       .00       .00       .00
Net Income Before Taxes                          -7.41      -.84      -.17       .30       .63       .64
 State Income Taxes                               -.37      -.04      -.01       .02       .03       .03
 Federal Income Tax                              -2.39      -.27      -.05       .10       .20       .21
 Federal Tax Credits                               .00       .00       .00       .00       .00       .00
Net Income After Taxes                           -4.65      -.53      -.10       .19       .40       .40
 plus Depreciation                                1.27      2.19      1.56      1.12       .80       .80
 plus Depletion                                    .00       .00       .00       .00       .00       .00
 less Depletable Items                             .00       .00       .00       .00       .00       .00
 less Depreciable/Capitalized Items               8.94       .00       .00       .00       .00       .00
 less Tax Credit on Expensable Items               .00       .00       .00       .00       .00       .00
Annual After Tax Cash Flow                      -12.31      1.66      1.46      1.31      1.19      1.20
Discounted After Tax Cash Flow                  -12.31      1.51      1.20       .98       .82       .74
Cumulative Discounted After Tax Cash Flow       -12.31    -10.80     -9.59     -8.61     -7.80     -7.05
```

Table B-16

Auxiliary OUTPUT File: 1493002.TCO

```
=====================================================================================

 PRIMARY WELLS ONLY, NO INFILLS

============================================================================

============================================================================


        Gross Gas     Original       Recovery
  Pay    Recovery   Gas-in-Place    Efficiency
 Grade    (MMcf)       (MMcf)        (% OGIP1)
 _____   _________   ____________   __________
   1       25454.       40669.         62.6%
   2       48815.       78005.         62.6%
   3       11310.       18076.         62.6%
 _____   _________   ____________   __________
 TOTAL     85579.      136750.         62.6%



 The Field Could No Longer Meet the Rate Constraint
 Beginning in Year 15.
\
============================================================================

 PRIMARY WELLS ONLY, NO INFILLS

============================================================================



                       PRODUCING RATE, MCFD/WELL

 # Wells   1.00     .00     .00    3.00     .00     .00    1.00     .00     .00    5.00

              Pay Grade 1             Pay Grade 2             Pay Grade 3
   Time ----------------------  ----------------------  ----------------------  Total
  years Well 1  Well 2  Well 3  Well 1  Well 2  Well 3  Well 1  Well 2  Well 3   Mcfd
 ------ ------  ------  ------  ------  ------  ------  ------  ------  ------  ------
  6.962  7571.      0.      0.   4990.      0.      0.   3474.      0.      0.   6169.
  7.962  2486.      0.      0.   1040.      0.      0.    562.      0.      0.   6169.
  8.962  2163.      0.      0.   1111.      0.      0.    672.      0.      0.   6169.
  9.962  2001.      0.      0.   1145.      0.      0.    732.      0.      0.   6169.
 10.962  1919.      0.      0.   1162.      0.      0.    764.      0.      0.   6169.

 11.962  1877.      0.      0.   1170.      0.      0.    781.      0.      0.   6169.
 12.962  1855.      0.      0.   1175.      0.      0.    790.      0.      0.   6169.
 13.962  1845.      0.      0.   1176.      0.      0.    795.      0.      0.   6169.
 14.962  1484.      0.      0.    923.      0.      0.    659.      0.      0.   4914.
 15.962   801.      0.      0.    485.      0.      0.    408.      0.      0.   2664.

 16.962   328.      0.      0.    217.      0.      0.    234.      0.      0.   1215.
 17.962   154.      0.      0.    115.      0.      0.    150.      0.      0.    649.
 18.962    69.      0.      0.     60.      0.      0.     95.      0.      0.    345.
 19.962    30.      0.      0.     32.      0.      0.     61.      0.      0.    186.
 20.962    13.      0.      0.     16.      0.      0.     39.      0.      0.    101.

 21.962     6.      0.      0.      9.      0.      0.     25.      0.      0.     56.
 22.962     0.      0.      0.      0.      0.      0.     16.      0.      0.     16.
 23.962     0.      0.      0.      0.      0.      0.     10.      0.      0.     10.
 24.962     0.      0.      0.      0.      0.      0.      6.      0.      0.      6.
 25.962     0.      0.      0.      0.      0.      0.      0.      0.      0.      0.

 26.962     0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
 27.962     0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
 28.962     0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
 29.962     0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
```

```
  30.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.

  31.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
  32.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
  33.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
  34.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
  35.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.

  36.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
  37.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
  38.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
  39.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
  40.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.

  41.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
  42.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
  43.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
  44.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.
  45.962      0.      0.      0.      0.      0.      0.      0.      0.      0.      0.

\
===============================================================================


 PRIMARY WELLS ONLY, NO INFILLS

===============================================================================



                     CUMULATIVE PRODUCTION, MMCF/WELL

 # Wells   1.00     .00     .00    3.00     .00     .00    1.00     .00     .00    5.00

               Pay Grade 1            Pay Grade 2            Pay Grade 3
    Time ----------------------  ----------------------  ----------------------  Total
   years Well 1  Well 2  Well 3  Well 1  Well 2  Well 3  Well 1  Well 2  Well 3   MMcf
  ------ ------  ------  ------  ------  ------  ------  ------  ------  ------  ------
   6.962  19237.      0.      0.  12681.      0.      0.   8828.      0.      0.  66108.
   7.962  20145.      0.      0.  13060.      0.      0.   9033.      0.      0.  68360.
   8.962  20934.      0.      0.  13466.      0.      0.   9279.      0.      0.  70611.
   9.962  21665.      0.      0.  13884.      0.      0.   9546.      0.      0.  72863.
  10.962  22365.      0.      0.  14308.      0.      0.   9825.      0.      0.  75115.

  11.962  23050.      0.      0.  14736.      0.      0.  10110.      0.      0.  77367.
  12.962  23727.      0.      0.  15164.      0.      0.  10398.      0.      0.  79618.
  13.962  24401.      0.      0.  15594.      0.      0.  10688.      0.      0.  81870.
  14.962  24943.      0.      0.  15931.      0.      0.  10929.      0.      0.  83664.
  15.962  25235.      0.      0.  16108.      0.      0.  11078.      0.      0.  84636.
```

Table B-17

Auxiliary OUTPUT File: 1493P002.PRR

```
                                           #
 GSAM ID      Tech.          P.G.  Dev. Case.         Resv.   OGIP  Wells  MASP
02314913P002  Current Technology   1 Primary *     7.0    40.7  1.  2.212     4.8
02314913P002  Current Technology   1 Refrac        7.0    40.7  1.  2.232     4.8
02314913P002  Current Technology   1 Infill       10.5    40.7  2.  2.420     9.3
02314913P002  Current Technology   2 Primary *    12.7    78.0  3.  3.889    14.5
02314913P002  Current Technology   2 Refrac       12.7    78.0  3.  3.920    14.5
02314913P002  Current Technology   2 Infill       19.5    78.0  6.  4.157    27.8
02314913P002  Current Technology   3 Primary *     2.9    18.1  1.  5.879     4.8
02314913P002  Current Technology   3 Refrac        2.9    18.1  1.  5.912     4.8
02314913P002  Current Technology   3 Infill        4.5    18.1  2.  6.266     9.3
02314913P002  Advanced Technology  1 Primary *     7.7    40.7  1.  2.020     4.4
02314913P002  Advanced Technology  1 Refrac        7.7    40.7  1.  2.039     4.4
02314913P002  Advanced Technology  1 Infill       11.0    40.7  2.  2.155     8.1
02314913P002  Advanced Technology  2 Primary *    15.0    78.0  3.  3.002    13.2
02314913P002  Advanced Technology  2 Refrac       15.0    78.0  3.  3.029    13.2
02314913P002  Advanced Technology  2 Infill       21.3    78.0  6.  3.193    24.4
02314913P002  Advanced Technology  3 Primary *     3.5    18.1  1.  4.319     4.4
02314913P002  Advanced Technology  3 Refrac        3.5    18.1  1.  4.350     4.4
02314913P002  Advanced Technology  3 Infill        5.0    18.1  2.  4.661     8.1
```

…..continued…..

```
@ $2/Mcf                                   @ $5/Mcf
Tot.   Undisc.  Disc.  Undisc.  Disc.   Tot.   Undisc.  Disc.  Undisc.  Disc.
 Cap.   AT Cash AT Cash BT Cash BT Cash  Cap.   AT Cash AT Cash BT Cash BT Cash
      .9     -.4     1.5      .3      .0      .0      .0      .0      .0
      .8     -.5     1.3      .3      .0      .0      .0      .0      .0
      .5    -1.1      .8     -.4      .0      .0      .0      .0      .0
    -3.3    -5.8    -5.3    -6.4      .0      .0      .0      .0      .0
    -3.6    -6.0    -5.7    -6.6      .0      .0      .0      .0      .0
    -8.4    -9.3   -13.4   -10.6      .0      .0      .0      .0      .0
    -2.1    -2.6    -3.3    -3.2      .0      .0      .0      .0      .0
    -2.1    -2.7    -3.4    -3.3      .0      .0      .0      .0      .0
    -4.4    -4.0    -7.0    -5.0      .0      .0      .0      .0      .0
     1.8     -.1     2.9      .9      .0      .0      .0      .0      .0
     1.8     -.1     2.8      .8      .0      .0      .0      .0      .0
     1.8     -.5     2.9      .5      .0      .0      .0      .0      .0
      .0    -3.7      .0    -3.1      .0      .0      .0      .0      .0
     -.3    -3.8     -.5    -3.3      .0      .0      .0      .0      .0
    -3.7    -5.9    -5.8    -5.8      .0      .0      .0      .0      .0
    -1.1    -2.0    -1.8    -2.2      .0      .0      .0      .0      .0
    -1.2    -2.0    -1.9    -2.2      .0      .0      .0      .0      .0
    -3.0    -2.9    -4.7    -3.4      .0      .0      .0      .0      .0
```

Table B-18

Auxiliary OUTPUT File: 1493P002.NPV

```
                             NPV Calculations
                             GSAM ID: 02314913P002 Tech.: Current Technology   Case: Primary P.G.:  1


                                                     Regular Case       +$1 Mcf          Zero Drill Cost     All Other
Costs Zero

NPV of cashflow ($MM)                                       -.4146            3.0031            3.1197             .1092

NPV Gas Prod. less Roy. and Sev. Tax (bcf)                  3.5426            3.5426            3.5426            3.5426
NPV Oil Prod. less Roy. and Sev. Tax (MMBbl)                 .0000             .0000             .0000             .0000
NPV of Gross Sales less Royalties ($MM)                     5.8276           14.5690            5.8276            5.8276
NPV of Expenses ($MM)                                       1.6792            3.2833            1.5540            1.0109
NPV of Tang. Investments (Excluding Drilling) ($MM)          .4673             .6776             .1651             .3172
NPV of Intang. Investments (Excluding Drilling) ($MM)        .0000             .0000             .0000             .0000
NPV of Development Well Costs ($MM)                         4.8601            7.4845             .0000            4.8601
NPV of Exploratory Well Costs ($MM)                          .0000             .0000             .0000             .0000
NPV of State and Federal Taxes ($MM)                         .4931            3.2644            2.2464             .7879
NPV of Depletable G&G/Lease ($MM)                            .0000             .0000             .0000             .0000
NPV of Expensed G&G/Lease ($MM)                              .0000             .0000             .0000             .0000
NPV of Federal Tax Credits ($MM)                             .0000             .0000             .0000             .0000

NPV of Project:                                            -1.6722            -.1410            1.8621           -1.1485
Total Cost of Proj:                                         6.5488            9.7876            1.2614            5.3025


Inc. in NPV WRT $1 Inc. in Gas Price:                    3.418
Inc. in NPV WRT 1 Million Dollar Drop in Drilling Cost:   .667
Inc. in NPV WRT 1 Million Dollar Drop in All Other Cost:  .415
```

Table B-19

OUTPUT File: *.ENV
This file is the environmental file output from the RP Module. It has been shortened to fit to one page in the Appendix. UNDISCP.ENV is shown below.

```
01116701P005    47  10003.       926.    .125  .00000  .00000 .00000        .000
01116701P006    47  10003.      1334.    .125  .00000  .00000 .00000        .000
01116701P007    47  10005.      1921.    .125  .00000  .00000 .00000        .000
01116701P008    47  10007.      2767.    .125  .00000  .00000 .00000        .000
01116701P009    47  10009.      3984.    .125  .00000  .00000 .00000        .000
01116701P010    47  10013.      5738.    .125  .00000  .00000 .00000        .000
01116701P011    47  10018.      8263.    .125  .00000  .00000 .00000        .000
01116701P012    47  10025.     11899.    .125  .00000  .00000 .00000        .000
01116701P013    47  10035.     17136.    .125  .00000  .00000 .00000        .000
01116701P014    47  10048.     24677.    .125  .00000  .00000 .00000        .000
01116701P015    47  10067.     35537.    .125  .00000  .00000 .00000        .000
01116701P016    47  10093.     51177.    .125  .00000  .00000 .00000        .000
01116701P017    47  10129.     73699.    .125  .00000  .00000 .00000        .000
01116702P005    37  10003.       926.    .125  .00000  .00000 .00000        .000
01116702P006    37  10003.      1334.    .125  .00000  .00000 .00000        .000
01116702P007    37  10005.      1921.    .125  .00000  .00000 .00000        .000
01116702P008    37  10007.      2767.    .125  .00000  .00000 .00000        .000
01116702P009    37  10009.      3984.    .125  .00000  .00000 .00000        .000
01116702P010    37  10013.      5738.    .125  .00000  .00000 .00000        .000
01116702P011    37  10018.      8263.    .125  .00000  .00000 .00000        .000
01116702P012    37  10025.     11899.    .125  .00000  .00000 .00000        .000
01116702P013    37  10035.     17136.    .125  .00000  .00000 .00000        .000
01116702P014    37  10048.     24677.    .125  .00000  .00000 .00000        .000
01116702P015    37  10067.     35537.    .125  .00000  .00000 .00000        .000
01116702P016    37  10093.     51177.    .125  .00000  .00000 .00000        .000
01116702P017    37  10129.     73699.    .125  .00000  .00000 .00000        .000
01116703P005    34  10003.       926.    .125  .00000  .00000 .00000        .000
01116703P006    34  10003.      1334.    .125  .00000  .00000 .00000        .000
01116703P007    34  10005.      1921.    .125  .00000  .00000 .00000        .000
01116703P008    34  10007.      2767.    .125  .00000  .00000 .00000        .000
01116703P009    34  10009.      3984.    .125  .00000  .00000 .00000        .000
01116703P010    34  10013.      5738.    .125  .00000  .00000 .00000        .000
01116703P011    34  10018.      8263.    .125  .00000  .00000 .00000        .000
01116703P012    34  10025.     11899.    .125  .00000  .00000 .00000        .000
01116703P013    34  10035.     17136.    .125  .00000  .00000 .00000        .000
01116703P014    34  10048.     24677.    .125  .00000  .00000 .00000        .000
01116703P015    34  10067.     35537.    .125  .00000  .00000 .00000        .000
01116703P016    34  10093.     51177.    .125  .00000  .00000 .00000        .000
```

Data Element	Description
1	GSAM ID
2	State code
3	Depth (ft.)
4	Area (acres)
5	Fraction of Federal lands
6	$C0_2$ content (fraction)
7	N_2 content (fraction)
8	H_2S content (fraction)
9	Condensate yield (Bbl/

APPENDIX C
EXPLORATION AND PRODUCTION MODULE FILES

CONTENTS

OUTPUT FILES

C-24	DECISION.OUT
C-25	PRICE.OUT
C-26	PRODSUMM.OUT
C-27	REVSUMM.OUT
C-28	SUPPSUMM.OUT
C-29	SUPPLY.EXT
C-30	UNRRSUMM.OUT
C-31	BNRRSUMM.OUT
C-32	RGRRSUMM.OUT
C-33	NRRSUMM.OUT
C-39	WELLSUMM.OUT
C-40	EXPLWLS.OUT

Table C-1

Input Specification File: SEQUEN.DAT (Location: \GSAM\EXPLPROD\NEWS)
This file specifies the files to be included in the construction of the environmental and processing files. NOTE: This file must follow the exact order and naming as in SPEC.DTU and in SPEC.DTD files respectively.

```
tmp\undiscff.env
tmp\undiscp.env
tmp\undtgtf.env
tmp\undtgtp.env
tmp\undcolf.env
tmp\undcolp.env
tmp\undofff.env
tmp\undcan.env
tmp\undchyp.env
tmp\undccn.env
tmp\appl.env
tmp\gsam01.env
tmp\gsam02.env
tmp\gsam03.env
tmp\gsam04.env
tmp\gsam05.env
tmp\gsam06.env
tmp\gsam07.env
tmp\gsam08.env
tmp\gsam09.env
tmp\gsam10.env
tmp\gsam11.env
tmp\gsam12.env
tmp\gsam15.env
tmp\gsam16.env
tmp\gsam17.env
tmp\gsam18.env
tmp\gsam99.env
tmp\canada.env
tmp\candu.env
```

Table C-2

Input Specification File: SPEC.DTD (Location: \GSAM\EXPLPROD\NEWS)
This file is used in creation of the discovered “data bank” files. It specifies the discovered resource by region to be included in the bank files. It must be copied to SPEC.DAT when used.

Location and names of output discovered banks

```
news\disb.bnk             news\disb.tcp
tmp\appl.dec                                  tmp\appl.prd
tmp\gsam01.dec                               tmp\gsam01.prd
tmp\gsam02.dec                               tmp\gsam02.prd
tmp\gsam03.dec                               tmp\gsam03.prd
tmp\gsam04.dec                               tmp\gsam04.prd
tmp\gsam05.dec                               tmp\gsam05.prd
tmp\gsam06.dec                               tmp\gsam06.prd
tmp\gsam07.dec                               tmp\gsam07.prd
tmp\gsam08.dec                               tmp\gsam08.prd
tmp\gsam09.dec                               tmp\gsam09.prd
tmp\gsam10.dec                               tmp\gsam10.prd
tmp\gsam11.dec                               tmp\gsam11.prd
tmp\gsam12.dec                               tmp\gsam12.prd
tmp\gsam15.dec                               tmp\gsam15.prd
tmp\gsam16.dec                               tmp\gsam16.prd
tmp\gsam17.dec                               tmp\gsam17.prd
tmp\gsam18.dec                               tmp\gsam18.prd
tmp\gsam99.dec                               tmp\gsam99.prd
tmp\canada.dec                               tmp\canada.prd
tmp\candu.dec                                 tmp\candu.prd
tmp\adgas.dec                                 tmp\adgas.prd
tmp\mex.dec                                  tmp\meximp.prd
```

Discovered and discovered producing file names to be used in creating discovered “data bank”

Table C-3

Input Specification File: SPEC.DTU (Location: \GSAM\EXPLPROD\NEWS)
This file is used in creation of the undiscovered "data bank" files. It specifies the undiscovered resource by region to be included in the bank files. It must be copied to SPEC.DAT when used.

Location and names of output undiscovered "bank" files

```
news\undb.bnk          news\undb.tcp
tmp\undiscf.dec                       tmp\undiscf.prd              tmp\undiscf.gsm
tmp\undiscp.dec                       tmp\undiscp.prd              tmp\undiscp.gsm
tmp\undtgtf.dec                       tmp\undtgtf.prd              tmp\undtgtf.gsm
tmp\undtgtp.dec                       tmp\undtgtp.prd              tmp\undtgtp.gsm
tmp\undcolf.dec                       tmp\undcolf.prd              tmp\undcolf.gsm
tmp\undcolp.dec                       tmp\undcolp.prd              tmp\undcolp.gsm
tmp\undofff.dec                       tmp\undofff.prd              tmp\undofff.gsm
tmp\undcan.dec                        tmp\undcan.prd               tmp\undcan.gsm
tmp\undchyp.dec                       tmp\undchyp.prd              tmp\undchyp.gsm
tmp\undccn.dec                        tmp\undccn.prd               tmp\undccn.gsm
```

Names of Undiscovered files to be used in creating undiscovered "data bank"

Table C-4

Input Data File: XXENV.DOE (Location: \GSAM\EXPLPROD\ENV)
This file is used in the creation of the environmental cost and processing files. It has information by year. The current example is for year 2005 (05ENV.DOE). Due to DOE's environmental R&D initiative environmental compliance costs are reduced from the status-quo base case scenario. This file contains the DOE R&D initiative costs. In this case the negative values for the costs represent the state-specific incremental reduction in costs compared to the base case for the environmental costs. All reservoirs located in a state would have the same incremental compliance costs. The file has been shortened to fit the length of the page.

```
               Existing   Existing   Existing        New        New        New
               Tangible Intangible        O&M   Tangible Intangible        O&M
State              Cost       Cost (Cum) Cost (Cum) Cost (Cum) Cost (Cum) Cost
Code              (K$)       (K$)       (K$)       (K$)       (K$)       (K$)
0100              .0000    28.9688     6.9573      .0000     -.4796     6.3535
0105              .0000     2.1437     -.5131      .0000    41.2192     -.6160
0110              .0000     6.9042     -.5131      .0000    41.2192     -.6160
0310              .0000     5.5445     -.7167      .0000    35.2548     -.7988
0350              .0000     5.5445     -.7167      .0000    35.2548     -.7988
0400              .0000    28.9688     6.9573      .0000     -.4796     6.3535
0405              .0000     1.2490     -.8888      .0000    21.9632     -.9477
0410              .0000     4.3999     -.8888      .0000    21.9632     -.9477
0450              .0000     4.3999     -.8888      .0000    21.9632     -.9477
0490              .0000    10.4003     -.8888      .0000    21.9632     -.9477
0900              .0000    28.9688     6.9573      .0000     -.4796     6.3535
0910              .0000     4.4746     -.8031      .0000    44.0434     -.8082
12                .0000     3.8765     -.8270      .0000    64.5092     -.8563
13                .0000     3.7355     -.7955      .0000    14.1517     -.8230
15                .0000     5.4687     -.7314      .0000    19.4434     -.8015
16                .0000     3.9993     -.9235      .0000    17.8683     -.9553
1700              .0000    28.9688     6.9573      .0000     -.4796     6.3535
1705              .0000     2.7621     -.6425      .0000    62.1567     -.7707
1710              .0000    12.3825     -.6425      .0000    62.1567     -.7707
1750              .0000     9.2474     -.6425      .0000    62.1567     -.7707
19                .0000      .0000      .0000      .0000      .0000      .0000
2                 .0000     8.7236     -.7067      .0000    32.2547     -.8926
21                .0000     7.7966     -.6974      .0000    12.2014     -.7585
2300              .0000    28.9688     6.9573      .0000     -.4796     6.3535
2310              .0000     8.9586     -.5821      .0000    66.4529     -.7991
24                .0000     5.0394     -.8083      .0000    26.7184     -.8351
25                .0000     4.8978     -.8089      .0000    13.5886     -.8507
26                .0000     4.7104     -.8098      .0000    38.9537     -.8525
27                .0000    15.2072     -.4201      .0000    51.1967     -.9462
3010              .0000     5.1099     -.7487      .0000    19.4018     -.8147
3050              .0000     5.1099     -.7487      .0000    19.4018     -.8147
31                .0000     3.8211     -.8659      .0000    15.8657     -.8962
33                .0000    15.6304     -.7653      .0000    44.6814     -.9471
34                .0000     3.8785     -.8812      .0000    13.0315     -.9114
```

.

.

.

(continues down with other states)

Table C-5

File: GASPRC.A99 (Location: \GSAM\EXPLPROD)
This file has gas price forecasts by region and by year of analysis. Shown below, it contains 1 price track (#1), but can be more (usually 5). The E&P module reads the file "GASPRC.NEW"; GASPRC.A99 should be copied into GASPRC.NEW before running the E&P module. **Note**: The file has been shortened to fit the width of this page.

```
Pacific Offshore     2  1   1.920   1.835   1.749   1.706   1.749   1.458   1.545   1.554
Pacific Onshore      2  1   1.920   1.835   1.749   1.696   1.752   1.534   1.704   1.759
```

```
San Juan             2  1   1.740   1.655   1.569   1.520   1.743   1.518   1.674   1.726
Rockies Foreland     2  1   1.730   1.610   1.489   1.519   1.743   1.518   1.674   1.726
Williston            2  1   1.650   1.535   1.419   1.450   1.743   1.518   1.674   1.726
Permian              2  1   1.810   1.720   1.629   1.588   2.056   1.722   1.851   1.917   Mid-Continent        2
1   1.740   1.660   1.579   1.529   2.194   1.924   2.062   2.092
Arkla-East Texas     2  1   1.850   1.760   1.669   1.617   2.187   1.896   2.046   2.110
Texas Gulf Coast     2  1   1.780   1.695   1.609   1.558   2.187   1.896   2.046   2.110   Gulf of Mexico-West  2
1   1.780   1.695   1.609   1.558   2.262   2.088   2.209   2.130   Gulf of Mexico-Cntr  2  1   1.820   1.740
1.659   1.608   2.262   2.088   2.209   2.130
Norphlet             2  1   1.840   1.755   1.669   1.627   2.262   2.088   2.209   2.130
Gulf of Mexico-East  2  1   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
So-Louisiana         2  1   1.850   1.765   1.679   1.637   2.187   1.896   2.046   2.110
MAFLA Onshore        2  1   1.800   1.715   1.629   1.588   2.187   1.896   2.046   2.110
Mid-West             2  1   1.870   1.785   1.699   1.648   2.375   1.982   2.072   2.057
Appalachia           2  1   1.960   1.880   1.799   1.755   2.557   2.041   2.082   2.023
Alberta              2  1   1.850   1.765   1.679   1.637   2.187   1.896   2.046   2.110
British Columbia     2  1   1.850   1.765   1.679   1.637   2.187   1.896   2.046   2.110
North Alaska         2  1   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
MacKenzie Delta      2  1   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Atlantic Offshore    2  1   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Mexico-Supply        2  1   1.800   1.720   1.639   1.529   1.209   1.472   1.601   1.667
```

Description of File: GASPRC.A99

Data Element	Description	Format
1	Supply region name	A20
2	Temporary index, not currently used	I3
3	Gas price track number	I3
4-...	Input wellhead (supply) price for time periods as specified in GEN_TML.SPC file ($/Mcf) 1X, F7.3	

(i.e. data element 4 for year 1993, data element 5 for year 1994, data element 6 for year 1995, data element 7 for year 1996, data element 8 for year 1997 and so on. Data elements 4,5,6, and 7 are for historical purposes only and the E&P Module does not operate with the years in them.).

Intended use of GASPRC.A99

For the D&I module, GASPRC.NEW (which GASPRC.A99 gets copied to) is an intermediate file. The final pass of a full integrated run does produce a final gas price file that shows the run's resulting gas prices. Note that the price tracks represent points on the supply curve, and that the second and fifth tracks show the equilibrium prices.

Data Elements of the file

The GASPRC.A99 file consists of four sets of data. The first set (the first column) has the name of the GSAM region. The second set is a column with a temporary index number that could be used as a state identifier later. The third set consists of a column with the price track number. The fourth set contains columns with the prices for each region. The number of columns depends on the number of years for which the forecast is reported. This set (matrix) of prices corresponds to a given price track. The time periods (the number of forecast years) are specified in the GEN_TML.SPC file. The different price tracks generate different supply curves for an integrated run of the model. Each supply curve is determined by several data points obtained from the forecasts for the gas prices and the levels of production corresponding to each of them.

The GASPRC.NEW file can also be derived from a randomly specified price track (for a sensitivity analysis). In this case it is a copy of the GASPRC.STR file, which has the chosen price track (the user-specified prices for each region for the time horizon specified in GEN_TML.SPC).

The supply curves (with multiple tracks) generated by this process are fed to the Demand and Integrating Modules of GSAM, where the equilibrium set of prices (the set of dual prices) is reached and returned to the Exploration and Production Module. There this set of prices is directed to the Production and Accounting Module.

Data Source Descriptions/Concerns

The GASPRC.A99 file is created by using the forecasts of EIA[1] obtained from its Annual Energy Outlook (AEO) publication. In this publication the prices for each supply region (as per EIA's NEMS model specification) are reported for a specific time horizon. This data is available through the EIA website in table 80 of the AEO's supplement[2]. The prices in the GSAM region Midwest are calculated as the average of the prices of the NEMS regions Northeast and Midcontinent. The historical data (for 1993-1996) for the Canadian regions in the model is obtained from CAPP. Since CAPP does not contain forecasts, the forecasts for Alberta and British Columbia prices are based on the basin price differentials with Henry Hub. We use the basin differential between the GSAM region South Louisiana (where Henry Hub is) and Alberta and subtract it from the forecast price for South Louisiana to obtain the forecast price for Alberta. This basin differential is forecasted to be $0.25 during the period 1999-2020. In this way the price in Alberta is always $0.25 lower than the price in South Louisiana throughout that period. We use the basin differential between Alberta and British Columbia and subtract it from the forecast price for Alberta to obtain the forecast price for British Columbia. This basin differential is $0.34 in 1997; it is forecasted to decrease linearly from $0.20 to $0.05 cents during the period 1998-2005 and stay constant at $0.05 thereafter. In this way the price in British Columbia is always lower (always $0.05 lower after 2005) than the price in Alberta.

[1] Energy Information Administration.

[2] http://www.eia.doe.gov/oiaf/supplement/sup99g.pdf

Table C-6

Input Data File: XXDOE.DOE (ENV3.OUT) (Location: \GSAM\EXPLPROD)
This file is output from the environmental file creation process (by ENV_WRTE.EXE), and input to the E&P Module. It has been shortened to fit to one page in the Appendix. It contains exactly the same information as in the XXDOE.ENV files, but is written in sparse format. Entries with a value of zero are eliminated from the XXDOE.ENV file.

GSAM ID

Number of non-zero elements (in XXDOE.ENV file) for environmental costs

```
01116701F005  3
 2       3.2340
 3       -.8332
 6       -.8332
01116701F006  3
 2       3.2340
 3       -.8332
 6       -.8332
01116701F007  3
 2       3.2340
 3       -.8332
 6       -.8332
01116701F008  3
 2       3.2340
 3       -.8332
 6       -.8332
01116701F009  3
 2       3.2340
 3       -.8332
 6       -.8332
01116701F010  3
 2       3.2340
 3       -.8332
 6       -.8332
```

Value for category specified

Corresponds to data element number in XXDOE.ENV file (i.e., Data Element 2 is existing intangible cost ($K), etc.)

Table C-7

Input Data File: DRL_CAP.SPC (Location: \GSAM\EXPLPROD)
This file contains various drilling specifications.

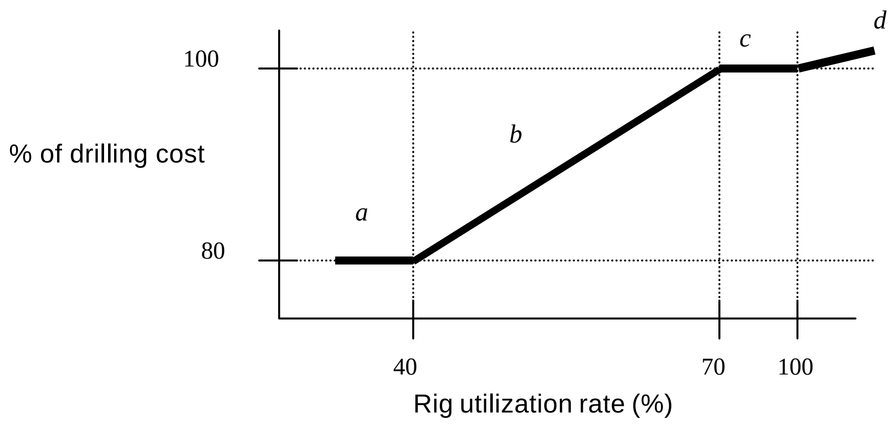
```
1.2     !  Drilling Efficiency (raises drilling by factor specified)
5.0     !  Rig Retirement Rate
80.0    !  Lowest % of to which the drilling cost can fall (i.e. Variable Drilling Cost)
100.0   !  % Drilling Cost in first EP year / AVG. Drilling Cost in RP
40.0    !  Minimum Variable Rig Utilization Rate (%)
70.0    !  Rig Utilization Rate (%) at Which Variable Drilling Cost Begin
110.0   !  Maximum Change in Rig Fleet (105 means 5% increase)
1.0     !  Annual Reduction in Drilling Cost (%)
2       !  Minimum number of wells to be drilled in a reservoir in U.S.
5.0     !  Maximum % of total wells that could be allowed in a year in U.S.
4.0     !  Minimum number of wells to be drilled in a reservoir in Canada
10.0    !  Maximum % of total wells that could be allowed in a year in Canada
```

<u>Explanation</u>

Drilling efficiency is defined in GSAM's E&P module as the percentage gain in efficiency when rigs move from exploration to development drilling (or loss in vice versa). For example: 1.2 means that when rigs move from exploration to development, their drilling capacity increases by 20% and conversely when rigs move from development to exploration their capacity decreases by 17% (1 - 1/1.2).

The **rig retirement rate** is the percent of drilling capacity (or rigs) that get retired every year.

The regional **rig utilization rate** drives the drilling cost as follows. As rig utilization, as a percent, increases, the costs of using rigs also increases because of the increased demand for rigs, up to the full drilling cost. The following graph shows this phenomenon:



Note that the **minimum variable rig utilization rate** is 40% and the **rig utilization rate at which the variable drilling cost begins** is 70%. When the regional rig utilization rate is less than 40%, low demand for rigs drives down the drilling costs. The drilling costs, calculated for development wells in the Reservoir Performance Module and for exploration wells in the E&P Module are subjected only to variable drilling costs and are therefore reduced by a factor. This factor is 80%. This corresponds to being on section *a* of the curve. When the regional rig utilization rate is above 70%, full drilling costs prevail, that is, fully 100% of the drilling costs calculated for development wells in the Reservoir Performance Module and for exploration wells in the E&P Module are applied. This would correspond to section *c* of the curve. When the regional rig utilization rate is between 40% and 70%, corresponding to section *b* on the curve, a linear interpolation between the drilling cost factors is performed (between 80% and 100%) to calculate a drilling cost factor in accordance with the rig utilization rate. The regional rig utilization rate may exceed 100%. This phenomenon of “super demand” can occur when rigs are moved into the high-demand region. Because moving drilling rigs is costly, the drilling cost factor will rise above 100% of the full drilling costs, corresponding to section *d* of the curve. Note that GSAM does not explicitly model individual drilling rigs, rather it models regional drilling capacity in terms of footage drilled.

The **maximum change in the rig fleet** indicates how much construction of capacity can occur in one year.

The **Minimum number of wells to be drilled in a reservoir** and the **Maximum % of total wells that could be allowed in a year in a reservoir** are used in computing the number of wells drilled in a year. For example, 5% maximum of total wells in a year would deplete the reservoir in 20 years if this maximum is achieved. There are separate factors for U.S and Canada.

<u>Intended Uses of DRL_CAP.SPC</u>

This file can be used to model drilling cost decline (an aggressive technology case could have a 3% decline percentage) or rig efficiency gains.

Table C-8

File: DRL_CST.SPC (Location: \GSAM\EXPLPROD)
This file contains entries to calculate EXPLORATORY drilling well cost (in $k) as a function of depth.

```
 1  1000.0      1.848788e-3   8.429423e-6  2.456291e-10   1.2    Pacific Offshore
 2  200.0000   -2.418747e-2   2.630253e-5 -6.552413e-10   1.2    Pacific Onshore
 3  54.40730    5.387449e-2  -7.106254e-6  1.339750e-9    1.2    San Juan
 4  47.21670    5.970161e-2  -6.851103e-6  7.622377e-10   1.2    Rockies Foreland
 5  17.8909     3.34051e-2    1.13753e-6   0.0            1.2    Williston
 6  35.65350    8.322879e-2  -1.829027e-5  1.632399e-9    1.2    Permian
 7  78.07079    1.775666e-2   1.481531e-6  3.708914e-10   1.2    Mid-Continent
 8  46.55651   -1.905254e-2   1.430119e-5 -3.249473e-10   1.2    Arkla-East Texas
 9  21.7314     1.848788e-3   8.429423e-6  2.456291e-10   1.2    Texas Gulf Coast
10  2239.81    -4.69329e-2    2.14607e-5   3.03929e-10    1.2    Gulf of Mexico-West
11  1715.511    3.880667e-1  -2.868301e-5  1.556441e-9    1.2    Gulf of Mexico-East
12  1715.511    3.880667e-1  -2.868301e-5  1.556441e-9    1.2    Norphlet
13  59.25750    5.948449e-2  -3.611307e-6  3.975062e-10   1.2    West Florida
14  299.0901    1.280414e-1  -1.890103e-5  1.784502e-9    1.2    So-Louisiana
15  59.25750    5.948449e-2  -3.611307e-6  3.975062e-10   1.2    MAFLA Onshore
16  34.16550    9.042406e-2  -3.209655e-7  9.641927e-10   1.2    Mid-West
17  27.068800   4.7098399e-2 -2.547277e-6  1.18087525e-10 1.2    Appalachia
20  300.0       1.848788e-3   8.429423e-6  2.456291e-10   1.2    North Alaska
21  78.07079    1.775666e-2   1.481531e-6  3.708914e-10   1.2    MacKenzie Delta
18  17.8909     3.34051e-2    1.13753e-6   0.0            1.7    Alberta
19  17.8909     3.34051e-2    1.13753e-6   0.0            2.21   British Columbia
22  1715.511    3.880667e-1  -2.868301e-5  1.556441e-9    1.2    Atlantic offshore
23  78.07079    1.775666e-2   1.481531e-6  3.708914e-10   1.2    Mexico-Supply
```

Description of File: DRL_CST.SPC

Data Element	Description	Format
1	Region identifier (different from Reservoir Performance Module's region number**	Free format integer
2	First coefficient (intercept) of the drilling cost equation	Free format integer
3	Second coefficient of the fourth order drilling cost equation	Free format integer
4	Third coefficient of the fourth order drilling cost equation	Free format integer
5	Fourth coefficient of the fourth order drilling cost equation	Free format integer
6	Drilling cost factor (for exploration wells	Free format real
7	Region name	No format

**Should be the same as in the NODE.SPC file

Explanation of DRL_CST.SPC

This file specifies the *exploration* drilling cost as a function of the development drilling cost. Remember that the development drilling costs have been calculated in the Reservoir Performance Module from the COST.DAT file, and that the coefficients specified here are the same as those development drilling costs in the RP Module. As such, these numbers are the coefficients of a polynomial regression equation that estimates the relationship between depth and drilling cost. The right-hand-side factors are the multiples for calculating the exploratory drilling cost from the development drilling cost. The drilling cost coefficients and intercepts provide exploration drilling cost in thousand dollars as a function of depth. The first cost column is the intercept and the next three are the coefficients of x^a where x is depth, in feet. The exploration drilling cost calculation is demonstrated in an example as follows:

For Appalachia the cost columns are: 27.068800 4.7098399e-2 -2.547277e-6 1.18087525e-10
and the exploration drilling factor is 1.2
So that: cost = $(27.07 + (4.71e^{-2})x + (-2.55e^{-6})x^2 + (1.18e^{-10})x^3)\ (1.2)$
If depth x = 1000 feet then:

$$\text{cost} = (27.07 + (4.71e^{-2})(1000) + (-2.55e^{-6})(1000)^2 + (1.18e^{-10})(1000)^3)\ (1.2) = 86.086$$

From these calculations, the cost of an exploration well in Appalachia, at a depth of 1000 feet is $86,086.

Intended Uses of DRL_CST.SPC

The coefficient numbers could be changed based on research of the historical/projected relation between depth and cost. The right-hand-side factors may be altered to take into account exploratory drilling that is more or less expensive in a particular region, or to test sensitivity of this parameter. It should be realized that these coefficients don't affect development drilling costs.

Table C-9

Input Data File: DRL_RCP.SPC (Location: \GSAM\EXPLPROD)
This file contains drilling rig capacity specifications.

```
Pacific Offshore       5.0     5.0    100.0     10.0      0
Pacific Onshore      146.9   298.2    200.0     10.0      1
San Juan              74.8   604.9    132.0     10.0      1
Rockies Foreland     709.0  5199.2    123.0     10.0      1
Williston             32.6    88.1    194.0     10.0      1
Permian              804.4  4941.5    128.0     10.0      1
Mid-Continent       1836.4 13467.2    113.0     10.0      1
Arkla-East Texas    1514.1  9300.8    114.0     10.0      1
Texas Gulf Coast    2497.8 12195.1    119.0     10.0      1
Gulf of Mexico-West  402.6   855.6    136.0     10.0      0
Gulf of Mexico-Cntr  976.4  2929.1    152.0     10.0      0
Norphlet               5.0     5.0    100.0     10.0      1
Gulf of Mexico-East    5.0     5.0    100.0     10.0      0
So-Louisiana         462.9  2623.2    112.0     10.0      1
MAFLA Onshore        320.3   865.9    139.0     10.0      1
Mid-West             269.6   955.9    117.0     10.0      1
Appalachia           306.2  7349.8    116.0     10.0      0
Alberta             6606.7  8757.7    145.0     10.0      1
British Columbia     884.2   816.2    145.0     10.0      1
North Alaska           5.0     5.0    100.0     10.0      1
MacKenzie Delta        5.0     5.0    100.0     10.0      1
Atlantic Offshore      5.0     5.0    100.0     10.0      0
Mexico-Supply          5.0     5.0    100.0     00.0      0
```

Description of File: DRL_RCP.SPC

Data Element	Description	Format
1	Supply region name	A20
2	Starting drilling rig capacity (in thousand ft) for exploration	F7.1, 1x
3	Starting drilling rig capacity (in thousand ft) for development	F7.1, t40
4	Maximum net increase in footage in a year(122 means 22% net increase)	F9.0, t50
5	Percentage of full drilling cost	F9.0, t60
6	Rig movement factor (a value of “0” means can not move rigs’ capacity out of region)	F9.0

Data Elements of DRL_RCP.SPC File

GSAM uses the data for the drilling footage capacity of development and exploratory wells to determine the level of drilling activity by region and year. This data is specified in the file DRL_RCP.SPC of the E&P module. The file consists of six sets of data, presented in six columns. The first column has GSAM region name (such as Appalachia, Gulf of Mexico-West etc.). The next two columns contain the total footage (in thousand ft.) for exploration and development gas wells respectively. The fourth column is maximum net percentage increase in footage per year. The fifth column contains the percentage of full drilling cost to move the rig into the respective region. The last column is an on/off switch with a value of "0" if rigs can not move out of the region (such as in Gulf of Mexico regions) and a value of "1" if it can. The user of the model can create different hypothetical cases by modifying the data in any of the columns of the file.

Summary

The drilling footage capacity is a parameter, used in GSAM for estimating future drilling activities in each region of the model. In the model it is represented by the total footage of exploratory and development gas wells drilled in each region. The maximum net increase of the drilling capacity is approximated as the maximum year-to-year percentage change in the drilling capacity for the period 1992-1997. Various data sources are used to complete the footage data.

Table C-10

File: DTEC_PEN.SPC (Location: \GSAM\EXPLPROD)
This file contains development technology penetration rates. It has been shortened to fit to one page in the Appendix. All years of technology penetration for a specific resource type are specified before going to another resource type.

```
Cur(dev) - Conv     1997 100.0 100.0
Cur(dev) - Conv     2000 100.0  86.0
Cur(dev) - Conv     2005 100.0  86.0
Cur(dev) - Conv     2010 100.0  66.0
Cur(dev) - Conv     2015 100.0  56.0
Cur(dev) - Conv     2020 100.0  46.0
Cur(dev) - Tight    1997 100.0 100.0
Cur(dev) - Tight    2000 100.0  86.0
Cur(dev) - Tight    2005 100.0  86.0
Cur(dev) - Tight    2010 100.0  66.0
Cur(dev) - Tight    2015 100.0  56.0
Cur(dev) - Tight    2020 100.0  46.0
Cur(dev) - Rad Flow 1997 100.0 100.0
Cur(dev) - Rad Flow 2000 100.0  86.0
Cur(dev) - Rad Flow 2005 100.0  86.0
Cur(dev) - Rad Flow 2010 100.0  66.0
Cur(dev) - Rad Flow 2015 100.0  56.0
Cur(dev) - Rad Flow 2020 100.0  46.0
Cur(dev) - Lin Flow 1997 100.0 100.0
Cur(dev) - Lin Flow 2000 100.0  86.0
Cur(dev) - Lin Flow 2005 100.0  86.0
Cur(dev) - Lin Flow 2010 100.0  66.0
Cur(dev) - Lin Flow 2015 100.0  56.0
Cur(dev) - Lin Flow 2020 100.0  46.0
Cur(dev) - W Drive  1997 100.0 100.0
Cur(dev) - W Drive  2000 100.0  86.0
Cur(dev) - W Drive  2005 100.0  86.0
Cur(dev) - W Drive  2010 100.0  66.0
Cur(dev) - W Drive  2015 100.0  56.0
Cur(dev) - W Drive  2020 100.0  46.0
Cur(dev) - Unconv   1997 100.0 100.0
Cur(dev) - Unconv   2000 100.0  86.0
Cur(dev) - Unconv   2005 100.0  86.0
Cur(dev) - Unconv   2010 100.0  66.0
Cur(dev) - Unconv   2015 100.0  56.0
Cur(dev) - Unconv   2020 100.0  46.0
Cur(dev) - Analyzed 1997 100.0 100.0
Cur(dev) - Analyzed 2000 100.0  86.0
Cur(dev) - Analyzed 2005 100.0  86.0
Cur(dev) - Analyzed 2010 100.0  66.0
Cur(dev) - Analyzed 2015 100.0  56.0
Cur(dev) - Analyzed 2020 100.0  46.0
Adv(dev) - Conv     1997  40.0 100.0
```

Data Element	Description	Format
1	Development technology parameter name	A20
2	Time period	I4
3	Technology penetration rate (%)	F6.1
4	Non-drilling cost factor	F6.1

Explanation of DETC PEN.SPC

This file shows the market penetration rate for development technology through time for current and advanced technology. Generally, current technology penetrates more and earlier than advanced technology.

A 100 value for the non-drilling cost factor means that operating and other non-drilling costs are the same as specified in the Reservoir Performance Module. The cost factor can be used to reduce (or increase) the non-drilling cost over time.

Development technology affects both the decision to explore (because the economics of exploration are directly related to the cost and recovery efficiencies of ultimate development practices) and the rate of development of a given resource. Hence the exploitation of undiscovered resource depends both on exploration and development technology penetration rates and the amount developed is constrained by the development technology penetration rates. For the discovered producing category development technology penetration curves affect the infill drilling and completion opportunities.

Intended Uses of DETC PEN.SPC

Development technology penetration rates are used in GSAM to reflect operator acceptance over time of emerging technologies. Technology impacts can be delayed (by setting a year's penetration rate to 0 or a very low value) to reflect the time required to develop, test, and implement new practices. Further, technology penetration for a given technology and resource can be flat or declining to reflect market saturation of a technology or to force a switch to an emerging technology from a less efficient method. The non-drilling costs can be varied to model the higher costs associated with initial applications of a given practice and the trend of falling costs with time as the technology is more widely understood and applied.

The application of any technology modeled in the Reservoir Performance Module can be varied using development technology penetration rates. This includes changes in skin factors, alternative hydraulic fracturing methods, lower (or higher) drilling costs, changes in completion methods, and alternative operating practices and costs.

DTEC_PEN.SPC can be used to measure the impact of increasing or decreasing a development technology's market penetration at any point in time. The cost factor parameter can be used to study the decrease/increase of non-drilling cost as a function of time. It should be realized that these penetration curves are applied to all features of development technology modeled in the Reservoir Performance Module.

Table C-11

Input Data File: DVL_TPR.SPC (Location: \GSAM\EXPLPROD)
This file defines development technology parameters and maps technologies to reservoir types.

```
Cur(dev) - Conv       1     0
Cur(dev) - Tight      2     0
Cur(dev) - Rad Flow   3     0
Cur(dev) - Lin Flow   4     0
Cur(dev) - W Drive    5     0
Cur(dev) - Unconv     6     0
Cur(dev) - Analyzed   7     0
Adv(dev) - Conv       1     1
Adv(dev) - Tight      2     1
Adv(dev) - Rad Flow   3     1
Adv(dev) - Lin Flow   4     1
Adv(dev) - W Drive    5     1
Adv(dev) - Unconv     6     1
Adv(dev) - Analyzed   7     1
```

Flag for current (= 0) or advanced (= 1)

Resource type
(currently, analyzed resource type is modeled for Gulf of Mexico)

Development technology parameter name

Table C-12

Input Specification File: ENV_DAT.SPC (Location: \GSAM\EXPLPROD)
This file defines the environmental files to be used in the E&P run.

# of environmental regulations (one for each year)

```
23
1998 98DOE.DNE
1999 99DOE.DNE
2000 00DOE.DNE
2001 01DOE.DNE
2002 02DOE.DNE
2003 03DOE.DNE
2004 04DOE.DNE
2005 05DOE.DNE
2006 06DOE.DNE
2007 07DOE.DNE
2008 08DOE.DNE
2009 09DOE.DNE
2010 10DOE.DNE
2011 11DOE.DNE
2012 12DOE.DNE
2013 13DOE.DNE
2014 14DOE.DNE
2015 15DOE.DNE
2016 16DOE.DNE
2017 17DOE.DNE
2018 18DOE.DNE
2019 19DOE.DNE
2020 20DOE.DNE
```

File containing environmental regulatory costs

Year in which environmental regulations become effective
(A maximum of 25 years of incremental regulations could be specified)

Table C-13

Input Data File: ENV_PROC.SPC (Location: \GSAM\EXPLPROD)
This file contains gas processing costs. It is output from the environmental file creation process (ENV_WRTE.EXE). It has been shortened to fit to one page. The ENV_PROC.OUT file is copied into ENV_PROC.SPC before running the E&P module. Those two files should have exactly the same number of entries in the same sequence.

GSAM ID

Effective gas processing cost in $/MCF for 1997 and 2020 respectively (calculated by subtracting the revenues from by-products from the impurities processing costs)

```
01116701F005       .0000       .0000
01116701F006       .0000       .0000
01116701F007       .0000       .0000
01116701F008       .0000       .0000
01116701F009       .0000       .0000
01116701F010       .0000       .0000
01116701F011       .0000       .0000
01116701F012       .0000       .0000
01116701F013       .0000       .0000
01116701F014       .0000       .0000
01116701F015       .0000       .0000
01116701F016       .0000       .0000
01116701F017       .0000       .0000
01116702F005       .0000       .0000
01116702F006       .0000       .0000
01116702F007       .0000       .0000
01116702F008       .0000       .0000
01116702F009       .0000       .0000
01116702F010       .0000       .0000
01116702F011       .0000       .0000
01116702F012       .0000       .0000
01116702F013       .0000       .0000
01116702F014       .0000       .0000
01116702F015       .0000       .0000
01116702F016       .0000       .0000
01116702F017       .0000       .0000
01116703F005       .0000       .0000
01116703F006       .0000       .0000
01116703F007       .0000       .0000
01116703F008       .0000       .0000
01116703F009       .0000       .0000
01116703F010       .0000       .0000
```

•

•

•

(continues with other reservoirs)

Table C-14

Input Data File: ENV_STAT.SPC (Location: \GSAM\EXPLPROD)
This file contains (static) entries by GSAMID for all reservoirs to be processed. It has been shortened to fit to one page in the Appendix. The ENV_STAT.OUT file is copied into ENV_STAT.SPC before running the E&P module. Those two files should have exactly the same number of entries in the same sequence.

```
01116701F005  47  10003.00      926.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116701F006  47  10003.00     1334.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116701F007  47  10005.00     1921.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116701F008  47  10007.00     2767.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116701F009  47  10009.00     3984.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116701F010  47  10013.00     5738.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116701F011  47  10018.00     8263.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116701F012  47  10025.00    11899.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116701F013  47  10035.00    17136.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116701F014  47  10048.00    24677.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116701F015  47  10067.00    35537.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116701F016  47  10093.00    51177.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116701F017  47  10129.00    73699.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116702F005  37  10003.00      926.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116702F006  37  10003.00     1334.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116702F007  37  10005.00     1921.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116702F008  37  10007.00     2767.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116702F009  37  10009.00     3984.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116702F010  37  10013.00     5738.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116702F011  37  10018.00     8263.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116702F012  37  10025.00    11899.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116702F013  37  10035.00    17136.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116702F014  37  10048.00    24677.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116702F015  37  10067.00    35537.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116702F016  37  10093.00    51177.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116702F017  37  10129.00    73699.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116703F005  34  10003.00      926.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116703F006  34  10003.00     1334.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116703F007  34  10005.00     1921.00       .1250       .0000       .0000       .0000         .00         .00         .00
01116703F008  34  10007.00     2767.00       .1250       .0000       .0000       .0000         .00         .00         .00
```

Description of File: ENV_STAT.SPC

Data Element	Description	Format
1	GSAM ID	A12, 1x
2	4-digit state code (See index Table A-8)	I4, 1x
3	Depth (ft.)	F10.2, 1x
4	Area (acres)	F12.2, 1x
5	Royalty rate (fraction)	F12.4, 1x
6	CO2 content (fraction)	F12.4, 1x
7	N2 content (fraction)	F12.4, 1x
8	H2S content (fraction)	F12.4, 1x
9	Lease bonus ($/ft)	F12.2, 1x
10	Condensate yield (BBL/MCF)	F12.2, 1x
11	H2O Yield (Bbl/MCF)	F12.2

Table C-15

Input Data File: ETEC_PEN.SPC (Location: \GSAM\EXPLPROD)
This file contains exploration technology penetration rates. It has been shortened to fit to one page in the Appendix. Only Current Technology is shown here. All years of technology penetration for a specific resource type are specified before going to another resource type.

```
Cur(exp) - Conv     1997 100.0     1
Cur(exp) - Conv     1998 100.0     1
Cur(exp) - Conv     2000 100.0     1
Cur(exp) - Conv     2010 100.0     1
Cur(exp) - Conv     2015 100.0     1
Cur(exp) - Conv     2020 100.0     1
Cur(exp) - Tight    1997 100.0     2
Cur(exp) - Tight    1998 100.0     2
Cur(exp) - Tight    2000 100.0     2
Cur(exp) - Tight    2010 100.0     2
Cur(exp) - Tight    2015 100.0     2
Cur(exp) - Tight    2020 100.0     2
Cur(exp) - Rad Flow 1997 100.0     3
Cur(exp) - Rad Flow 1998 100.0     3
Cur(exp) - Rad Flow 2000 100.0     3
Cur(exp) - Rad Flow 2010 100.0     3
Cur(exp) - Rad Flow 2015 100.0     3
Cur(exp) - Rad Flow 2020 100.0     3
Cur(exp) - Lin Flow 1997 100.0     4
Cur(exp) - Lin Flow 1998 100.0     4
Cur(exp) - Lin Flow 2000 100.0     4
Cur(exp) - Lin Flow 2010 100.0     4
Cur(exp) - Lin Flow 2015 100.0     4
Cur(exp) - Lin Flow 2020 100.0     4
Cur(exp) - W Drive  1997 100.0     5
Cur(exp) - W Drive  1998 100.0     5
Cur(exp) - W Drive  2000 100.0     5
Cur(exp) - W Drive  2010 100.0     5
Cur(exp) - W Drive  2015 100.0     5
Cur(exp) - W Drive  2020 100.0     5
Cur(exp) - Unconv   1997 100.0     6
Cur(exp) - Unconv   1998 100.0     6
Cur(exp) - Unconv   2000 100.0     6
Cur(exp) - Unconv   2010 100.0     6
Cur(exp) - Unconv   2015 100.0     6
Cur(exp) - Unconv   2020 100.0     6
Cur(exp) - Analyzed 1997 100.0     7
Cur(exp) - Analyzed 1998 100.0     7
Cur(exp) - Analyzed 2000 100.0     7
Cur(exp) - Analyzed 2010 100.0     7
Cur(exp) - Analyzed 2015 100.0     7
Cur(exp) - Analyzed 2020 100.0     7
```

Description of File: ETEC_PEN.SPC

Data Element	Description	Format
1	Exploration technology parameter name	A20
2	Time period	I4
3	Exploration technology penetration rate (%)	F6.1
4	Resource type	I6

Explanation

ETEC_PEN.SPC is used to model the penetration curve of current and advanced exploration technology.

Exploration technology penetration affects only the decision to apply a given exploration dry hole rate and efficiency to find remaining undiscovered resources in given plays. Development technology penetration (in file DTEC_PEN.SPC), on the other hand, controls the used of alternative development methods on explored plays. The application of any technology modeled in the Reservoir Performance Module can be varied using *development* technology penetration rates. This includes changes in skin factors, alternative hydraulic fracturing methods, lower (or higher) drilling costs, changes in completion methods, and alternative operating practices and costs.

Intended Uses of ETEC_PEN.SPC

ETEC_PEN.SPC can be used to measure the impact of increasing or decreasing an exploration technology's market penetration at any point in time. Note that all of the resource types should be specified for a year. The values are interpolated (or extrapolated) for years in which data is not provided.

Table C-16

Input Data File: EXP_DFN.SPC (Location: \GSAM\EXPLPROD)

This file contains exploration definition specifications. It has been shortened to fit onto 1 page in the Appendix.

```
Cur(exp) - Conv     25.0   1.0000 0.5657 0.3200 0.1810 0.1024 0.0579 0.0328 0.0185 0.0131 0.0093 0.0066 0.0046 0.0033
Cur(exp) - Conv     25.0   0.0000 1.0000 0.5657 0.3200 0.1810 0.1024 0.0579 0.0328 0.0232 0.0164 0.0116 0.0082 0.0058
Cur(exp) - Conv     25.0   0.0000 0.0000 1.0000 0.5657 0.3200 0.1810 0.1024 0.0579 0.0410 0.0290 0.0205 0.0145 0.0102
Cur(exp) - Conv     25.0   0.0000 0.0000 0.0000 1.0000 0.5657 0.3200 0.1810 0.1024 0.0724 0.0512 0.0362 0.0256 0.0181
Cur(exp) - Conv     25.0   0.0000 0.0000 0.0000 0.0000 1.0000 0.5657 0.3200 0.1810 0.1280 0.0905 0.0640 0.0453 0.0320
Cur(exp) - Conv     25.0   0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.5657 0.3200 0.2263 0.1600 0.1131 0.0800 0.0566
Cur(exp) - Conv     25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.5657 0.4000 0.2828 0.2000 0.1414 0.1000
Cur(exp) - Conv     25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.5000 0.3536 0.2500 0.1768
Cur(exp) - Conv     25.0   0.0000 0.000                                      .0000 1.0000 0.7071 0.5000 0.2500 0.1768
Cur(exp) - Conv     25.0   0.0000 0.000                                      .0000 0.0000 1.0000 0.7071 0.3536 0.2500
Cur(exp) - Conv     25.0   0.0000 0.000                                      .0000 0.0000 0.0000 1.0000 0.5000 0.3536
Cur(exp) - Conv     25.0   0.0000 0.000                                      .0000 0.0000 0.0000 0.0000 1.0000 0.7071
Cur(exp) - Conv     25.0   0.0000 0.000                                      .0000 0.000
Cur(exp) - Tight    25.0   1.0000 0.404                                      .0018 0.000
Cur(exp) - Tight    25.0   0.0000 1.000                                      .0044 0.001
Cur(exp) - Tight    25.0   0.0000 0.000                                      .0108 0.004
Cur(exp) - Tight    25.0   0.0000 0.000                                      .0267 0.010
Cur(exp) - Tight    25.0   0.0000 0.000                                      .0660 0.026
Cur(exp) - Tight    25.0   0.0000 0.000                                      .1633 0.066
Cur(exp) - Tight    25.0   0.0000 0.000                                      .4041 0.163
Cur(exp) - Tight    25.0   0.0000 0.0000                                     1.0000 0.404
Cur(exp) - Tight    25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.000
Cur(exp) - Tight    25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.000
Cur(exp) - Tight    25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.000
Cur(exp) - Tight    25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.000
Cur(exp) - Tight    25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000
Cur(exp) - Rad Flow 25.0   1.0000 0.4041 0.1633 0.0660 0.0267 0.0108 0.0044 0.0018 0.0007 0.0005 0.0004 0.0003 0.0002
Cur(exp) - Rad Flow 25.0   0.0000 1.0000 0.4041 0.1633 0.0660 0.0267 0.0108 0.0044 0.0018 0.0012 0.0009 0.0006 0.0004
Cur(exp) - Rad Flow 25.0   0.0000 0.0000 1.0000 0.4041 0.1633 0.0660 0.0267 0.0108 0.0044 0.0031 0.0022 0.0015 0.0011
Cur(exp) - Rad Flow 25.0   0.0000 0.0000 0.0000 1.0000 0.4041 0.1633 0.0660 0.0267 0.0108 0.0076 0.0054 0.0038 0.0027
Cur(exp) - Rad Flow 25.0   0.0000 0.0000 0.0000 0.0000 1.0000 0.4041 0.1633 0.0660 0.0267 0.0188 0.0133 0.0094 0.0067
Cur(exp) - Rad Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.4041 0.1633 0.0660 0.0466 0.0330 0.0233 0.0165
Cur(exp) - Rad Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.4041 0.1633 0.1154 0.0816 0.0577 0.0408
Cur(exp) - Rad Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.4041 0.2857 0.2020 0.1429 0.1010
Cur(exp) - Rad Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.5000 0.3536 0.2500
Cur(exp) - Rad Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.5000 0.3536
Cur(exp) - Rad Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.5000
Cur(exp) - Rad Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071
Cur(exp) - Rad Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000
Cur(exp) - Lin Flow 25.0   1.0000 0.4041 0.1633 0.0660 0.0267 0.0108 0.0044 0.0018 0.0007 0.0005 0.0004 0.0003 0.0002
Cur(exp) - Lin Flow 25.0   0.0000 1.0000 0.4041 0.1633 0.0660 0.0267 0.0108 0.0044 0.0018 0.0012 0.0009 0.0006 0.0004
Cur(exp) - Lin Flow 25.0   0.0000 0.0000 1.0000 0.4041 0.1633 0.0660 0.0267 0.0108 0.0044 0.0031 0.0022 0.0015 0.0011
Cur(exp) - Lin Flow 25.0   0.0000 0.0000 0.0000 1.0000 0.4041 0.1633 0.0660 0.0267 0.0108 0.0076 0.0054 0.0038 0.0027
Cur(exp) - Lin Flow 25.0   0.0000 0.0000 0.0000 0.0000 1.0000 0.4041 0.1633 0.0660 0.0267 0.0188 0.0133 0.0094 0.0067
Cur(exp) - Lin Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.4041 0.1633 0.0660 0.0466 0.0330 0.0233 0.0165
Cur(exp) - Lin Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.4041 0.1633 0.1154 0.0816 0.0577 0.0408
Cur(exp) - Lin Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.4041 0.2857 0.2020 0.1429 0.1010
Cur(exp) - Lin Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.5000 0.3536 0.2500
Cur(exp) - Lin Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.5000 0.3536
Cur(exp) - Lin Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.5000
Cur(exp) - Lin Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071
Cur(exp) - Lin Flow 25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000
Cur(exp) - W Drive  25.0   1.0000 0.5657 0.3200 0.1810 0.1024 0.0579 0.0328 0.0185 0.0131 0.0093 0.0066 0.0046 0.0033
Cur(exp) - W Drive  25.0   0.0000 1.0000 0.5657 0.3200 0.1810 0.1024 0.0579 0.0328 0.0232 0.0164 0.0116 0.0082 0.0058
Cur(exp) - W Drive  25.0   0.0000 0.0000 1.0000 0.5657 0.3200 0.1810 0.1024 0.0579 0.0410 0.0290 0.0205 0.0145 0.0102
Cur(exp) - W Drive  25.0   0.0000 0.0000 0.0000 1.0000 0.5657 0.3200 0.1810 0.1024 0.0724 0.0512 0.0362 0.0256 0.0181
Cur(exp) - W Drive  25.0   0.0000 0.0000 0.0000 0.0000 1.0000 0.5657 0.3200 0.1810 0.1280 0.0905 0.0640 0.0453 0.0320
Cur(exp) - W Drive  25.0   0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.5657 0.3200 0.2263 0.1600 0.1131 0.0800 0.0566
Cur(exp) - W Drive  25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.5657 0.4000 0.2828 0.2000 0.1414 0.1000
Cur(exp) - W Drive  25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.5000 0.3536 0.2500 0.1768
Cur(exp) - W Drive  25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.5000 0.2500 0.1768
Cur(exp) - W Drive  25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.3536 0.2500
Cur(exp) - W Drive  25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.5000 0.3536
Cur(exp) - W Drive  25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071
Cur(exp) - W Drive  25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000
Cur(exp) - Unconv   35.0   1.0000 0.2176 0.0473 0.0103 0.0022 0.0005 0.0001 0.0001 0.0001 0.0000 0.0000 0.0000 0.0000
Cur(exp) - Unconv   35.0   0.0000 1.0000 0.2176 0.0473 0.0103 0.0022 0.0005 0.0003 0.0002 0.0002 0.0001 0.0001 0.0001
Cur(exp) - Unconv   35.0   0.0000 0.0000 1.0000 0.2176 0.0473 0.0103 0.0022 0.0016 0.0011 0.0008 0.0006 0.0004 0.0003
Cur(exp) - Unconv   35.0   0.0000 0.0000 0.0000 1.0000 0.2176 0.0473 0.0103 0.0073 0.0051 0.0036 0.0026 0.0018 0.0013
```

Probability Weight for Field Size Class 17, 17 is largest FSC Available

Probability Weight for Field Size Class 5, 12 is largest FSC Available

Description of File: EXP_DFN.SPC

Data Element	Description	Format
1	Exploration technology definition name	A20
2	Exploration success rate (%)	F5.1, 2X
3-15	Exploration curve by field size class (17-5)	F7.4

Explanation of EXP DFN.SPC

The file EXP_DFN.SPC file is used in GSAM to model the uncertainty inherent in exploration practices. The uncertainty, in this case, concerns the probability of finding accumulations (reservoirs) in a specific field size class (FSC) of a play. Modeling this uncertainty is based on the premise that the remaining reservoirs that are undiscovered are of a size less than or equal to a maximum field size class already explored.

Given that a FSC is available, 15 or less in this example, the uncertainty arises in the chances of finding any reservoir; because reservoirs of FSC 15 are available, they are the most likely to be found in this case. The next most likely would be FSC 14, and so on down to FSC 5. According to this logic, a weight must be assigned to a reservoir size, indicating that it is more or less likely to be found, and this is the purpose of the EXP_DFN.SPC file's matrix. Each row in the matrix can be thought of as the **exploration curve** , in any year, for a given technology, for a given resource type, for a given FSC availability. As an example, Figure A shows the exploration curve for current exploration technology in conventional reservoirs with FSC 15 or less available (corresponding to the third row of the file's matrix).

Figure A

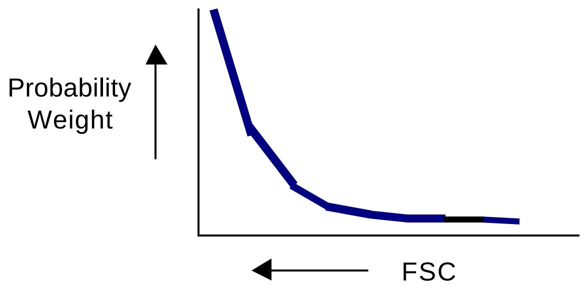
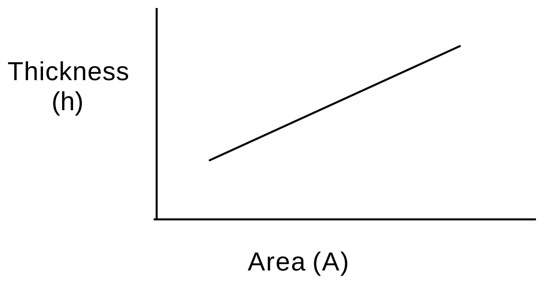


Figure B



Given that a FSC is available to be discovered, the relative weights on the probability of discovery are derived from one assumption: that the chance of finding a larger reservoir (one with a greater area) is better than the chances of discovering a smaller reservoir. Reservoirs that have a FSC of 10 or below are too small to differentiate them from one another, so that the weights on the chances of finding any of these reservoirs are based only on the area of the reservoirs in a FSC. When the area is the only factor that determines the probability of reservoir discovery, the discovery process is considered “random”. Reservoirs having a FSC of 17 to 11 also base their probability weights on area, but include an additional factor that takes into account the ease of differentiation among these larger reservoirs by using technology.

As field size classes increase, the volume in the FSC doubles (FSC 10 has an average recoverable reserves of 19.2 and FSC 11 has an average OGIP of 38.4) **(1)**. Also, the thickness and area are assumed to be linearly related, as seen in Figure B **(2)**. From these two pieces of information, the relative probability weights based on area can be derived. If reservoirs of FSC 10 are the largest remaining available, the relative weight on the probability of finding a reservoir of FSC 10 is 1. The relative weight on the probability of finding a reservoir of FSC 9 is less than that of FSC 10 by a multiplying factor. This factor is $(1/\sqrt{2})$, and is calculated by transforming the following ratio:

Note: Area*Thickness = Volume or Ah = OGIP *and* A_x is area in FSCx *and* h_x is thickness in FSCx

$A_9h_9 / A_{10}h_{10} = 1/2$ because of **(1)** above

$h = cA$ from **(2)** above; where c is constant of proportionality

$A_9(cA_9)/A_{10}(cA_{10}) = 1/2$

$A_9^2/A_{10}^2 = 1/2$

$A_9/A_{10} = 1/\sqrt{2}$

Therefore, if FSC 10 is the largest available, the weight on the chances of finding a reservoir in FSC 10 = 1, FSC 9 = 1(1/ √ 2) = 0.7071, FSC 8 = 1(1/ √ 2)(1/ √ 2) = 0.50, etc. For reservoirs in a FSC above 10, the same formula applies, with an additional factor. The model assumes that for these larger reservoirs, the chances of finding a reservoir in a FSC is 25% better than random for each larger FSC in conventional, water-drive, and offshore reservoirs and 75% better than random for each larger FSC in tight, radial and linear flow, and unconventional. So that if FSC 15 is the largest size of the remaining conventional reservoirs, the chance of finding a FSC 15 reservoir is assigned a weight of 1. For FSC 14 the weighted chance would be 1(1/ √ 2)(1/1.25) = 0.5667. For FSC 13 the value would be 1(1/ √ 2) (1/ √ 2)(1/1.25) = 0.3200, and so forth. For tight these values would be 1(1/ √ 2) (1/1.75) = 0.4041 and so on.

As the numbers in this exploration curve do not sum to one, they are not probabilities, but relative weights, as mentioned above. To transform the weights into probabilities, sum the values in an exploration curve, take the reciprocal, and multiply the reciprocal by each weights for the individual probabilities of finding a reservoir in a specific field size class. To calculate the overall probability of finding a reservoir of a certain FSC multiply each of the individual probabilities by the **exploration success rate**, in the second column, to incorporate the chance of drilling a dry hole in exploration. As an example, the probability of successfully finding and drilling a FSC 13 reservoir when the largest conventional reservoir available is size 15 is:

(Reciprocal of sum of weights) (Individual FSC weight) (Success rate)

(1 / 2.3422) (0.3200) (.14) = 0.01913 = 1.9% probability

Advanced technology will generally have a higher success rate than current.

<u>Intended Uses of EXP_DFN.SPC</u>

This file can be used in modeling the effects of better seismic technology, improving the resolution of smaller field size classes. This could be done by changing (increasing) the relative probability weights of smaller (less than 10) field size classes. The exploration success rate, could also be changed to model alternative scenarios.

Table C-17

Input Data File: EXP_PLY.SPC (Location: \GSAM\EXPLPROD)
This file contains detailed specifications for exploration risk by play, in the same format as EXP_DFN.SPC.

```
602  Cur(exp) - Conv     14.0   1.0000 0.5657 0.3200 0.1810 0.1024 0.0579 0.0328 0.0185 0.0131 0.0093 0.0066 0.0046 0.0033
602  Cur(exp) - Conv     14.0   0.0000 1.0000 0.5657 0.3200 0.1810 0.1024 0.0579 0.0328 0.0232 0.0164 0.0116 0.0082 0.0058
602  Cur(exp) - Conv     14.0   0.0000 0.0000 1.0000 0.5657 0.3200 0.1810 0.1024 0.0579 0.0410 0.0290 0.0205 0.0145 0.0102
602  Cur(exp) - Conv     14.0   0.0000 0.0000 0.0000 1.0000 0.5657 0.3200 0.1810 0.1024 0.0724 0.0512 0.0362 0.0256 0.0181
602  Cur(exp) - Conv     14.0   0.0000 0.0000 0.0000 0.0000 1.0000 0.5657 0.3200 0.1810 0.1280 0.0905 0.0640 0.0453 0.0320
602  Cur(exp) - Conv     14.0   0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.5657 0.3200 0.2263 0.1600 0.1131 0.0800 0.0566
602  Cur(exp) - Conv     14.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.5657 0.4000 0.2828 0.2000 0.1414 0.1000
602  Cur(exp) - Conv     14.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.5000 0.3536 0.2500 0.1768
602  Cur(exp) - Conv     14.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.5000 0.2500 0.1768
602  Cur(exp) - Conv     14.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.3536 0.2500
602  Cur(exp) - Conv     14.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.5000 0.3536
602  Cur(exp) - Conv     14.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071
602  Cur(exp) - Conv     14.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000
985  Cur(exp) - Tight    25.0   1.0000 0.4041 0.1633 0.0660 0.0267 0.0108 0.0044 0.0018 0.0007 0.0005 0.0004 0.0003 0.0002
985  Cur(exp) - Tight    25.0   0.0000 1.0000 0.4041 0.1633 0.0660 0.0267 0.0108 0.0044 0.0018 0.0012 0.0009 0.0006 0.0004
985  Cur(exp) - Tight    25.0   0.0000 0.0000 1.0000 0.4041 0.1633 0.0660 0.0267 0.0108 0.0044 0.0031 0.0022 0.0015 0.0011
985  Cur(exp) - Tight    25.0   0.0000 0.0000 0.0000 1.0000 0.4041 0.1633 0.0660 0.0267 0.0108 0.0076 0.0054 0.0038 0.0027
985  Cur(exp) - Tight    25.0   0.0000 0.0000 0.0000 0.0000 1.0000 0.4041 0.1633 0.0660 0.0267 0.0188 0.0133 0.0094 0.0067
985  Cur(exp) - Tight    25.0   0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.4041 0.1633 0.0660 0.0466 0.0330 0.0233 0.0165
985  Cur(exp) - Tight    25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.4041 0.1633 0.1154 0.0816 0.0577 0.0408
985  Cur(exp) - Tight    25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.4041 0.2857 0.2020 0.1429 0.1010
985  Cur(exp) - Tight    25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.5000 0.3536 0.2500
985  Cur(exp) - Tight    25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.5000 0.3536
985  Cur(exp) - Tight    25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071 0.5000
985  Cur(exp) - Tight    25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.7071
985  Cur(exp) - Tight    25.0   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000
```

Description of File: EXP_PLY.SPC

Data Element	Description	Format
1	Play number (4-digit play code)	A20
2	Technology parameter	A20
3	Exploration success rate (%)	F5.1, 2X
4-13	Exploration factors for all the available field size classes (17-5)	F7.4

<u>Explanation</u>

This file contains specifications for exploration risk by play, in the same format as EXP_DFN.SPC. Data in this play-specific file will supersede the resource type data in EXP_DFN.SPC. If EXP_PLY.SPC is missing or is of size 0, the EXP_DFN.SPC information will be used exclusively. See Table C-16 for an in-depth description of this file's function.

<u>Intended Uses</u>

This file is not required to run the E&P Module. However, if desired, the exploration risk and relative probability can be specified for particular plays.

Table C-18

Input Data File: GEN_TML.SPC (Location: \GSAM\EXPLPROD)

```
1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010
10       ! (Discount Rate, %)
2.00     ! (Screening Gas Price, $/MCF)
1997     ! (Start Year, tmex)
0        ! (Marginal Gas Rate, 60 MCF/Day/Well)
12.5     ! (Royalty Rate For Incentive Case, %)
0        ! (Enter 1: For PLAY Specific PRODSUMM File, Enter 0: Otherwise)
```

Explanation

The GASPRC.NEW file should have prices for the time periods specified above. The first year is the model starting run year. The marginal gas rate is specified when running royalty-incentive case runs, when the royalty rate is applied.

Table C-19

Input Data File: NODE.SPC (Location: \GSAM\EXPLPROD)
This file contains supply region specifications for plays.

```
New England          0  1   0.00  0.000
Middle Atlantic      0  2   0.00  0.000
South Atlantic       0  3   0.00  0.000
Florida              0  4   0.00  0.000
East South Central   0  5   0.00  0.000
East North Central   0  6   0.00  0.000
West South Central   0  7   0.00  0.000
West North Central   0  8   0.00  0.000
Mountain South       0  9   0.00  0.000
Mountain North       0 10   0.00  0.000
California           0 11   0.00  0.000
Pacific Northwest    0 12   0.00  0.000
Canada-East          0 13   0.00  0.000
Western Canada       0 14   0.00  0.000
BC-Demand            0 15   0.00  0.000
Mexico-Demand        0 16   0.00  0.000
Pacific Offshore     1  0   1.05  0.150
Pacific Onshore      2  0   1.05  0.150
San Juan             3  0   1.05  0.170
Rockies Foreland     4  0   1.05  0.120
Williston            5  0   1.05  0.150
Permian              6  0   1.05  0.100
Mid-Continent        7  0   1.05  0.100
Arkla-East Texas     8  0   1.05  0.070
Texas Gulf Coast     9  0   1.05  0.080
Gulf of Mexico-West 10  0   1.05  0.070
Gulf of Mexico-East 11  0   1.05  0.080
Norphlet            12  0   1.05  0.080
West Florida        13  0   1.05  0.080
So-Louisiana        14  0   1.05  0.070
MAFLA Onshore       15  0   1.05  0.150
Mid-West            16  0   1.05  0.150
Appalachia          17  0   1.05  0.120
North Alaska        20  0   1.05  0.150
MacKenzie Delta     21  0   1.05  0.150
Alberta             18  0   1.13  0.190
British Columbia    19  0   1.13  0.150
Sable Island         0  0   0.00  0.000
Distrigas            0  0   0.00  0.000
Cove Point           0  0   0.00  0.000
Elba Island          0  0   0.00  0.000
Lake Charles         0  0   0.00  0.000
Arctic Islands       0  0   0.00  0.000
Atlantic Offshore   22  0   1.05  0.150
Mexico-Supply       23  0   1.13  0.150
Alliance-Supply      0  0   0.00  0.000
```

Description of File: NODE.SPC

Data Element	Description	Format
1	Region name	A20
2	Supply region indicator	I2, 1X
3	Demand region indicator (read and used only in the Demand and Integrating Module)	I2, 1X
4	Supply load factor	F6.3
5	Gathering cost, $/Mcf	F6.3

Explanation of NODE.SPC

This file is used to relate the supply region names to a counter in the E&P Module.

Table C-20

Input Data File: PLY_DFN.SPC (Location: \GSAM\EXPLPROD)
This file contains play-level specifications. This file has been shortened to fit to one page in the Appendix.

```
USSG Play          GSAM Supply         #  Dev. Succ. Rate  Res. Expl.  Roy.
Code               Region                 Curr  n/a   Adv  Type Depth  Rate
0101F              North Alaska        1  80.0  90.0  90.0 1   7577.0  12.5
0101P              North Alaska        1  80.0  90.0  90.0 1   7577.0  12.5
0102F              North Alaska        1  80.0  90.0  90.0 1   9300.0  12.5
0102P              North Alaska        1  80.0  90.0  90.0 1   9300.0  12.5
0103F              North Alaska        1  80.0  90.0  90.0 1   7076.0  12.5
0103P              North Alaska        1  80.0  90.0  90.0 1   7076.0  12.5
0105F              North Alaska        1  80.0  90.0  90.0 1   9300.0  12.5
0105P              North Alaska        1  80.0  90.0  90.0 1   9300.0  12.5
0106F              North Alaska        1  80.0  90.0  90.0 1  10000.0  12.5
0106P              North Alaska        1  80.0  90.0  90.0 1  10000.0  12.5
0109F              North Alaska        1  80.0  90.0  90.0 1   1573.0  12.5
0109P              North Alaska        1  80.0  90.0  90.0 1   1573.0  12.5
0110F              North Alaska        1  80.0  90.0  90.0 1   9300.0  12.5
0110P              North Alaska        1  80.0  90.0  90.0 1   9300.0  12.5
0111F              North Alaska        1  80.0  90.0  90.0 1   6515.0  12.5
0111P              North Alaska        1  80.0  90.0  90.0 1   6515.0  12.5
0201F              North Alaska        1  80.0  90.0  90.0 1   9300.0  12.5
0201P              North Alaska        1  80.0  90.0  90.0 1   9300.0  12.5
0205F              North Alaska        1  80.0  90.0  90.0 1   9300.0  12.5
0205P              North Alaska        1  80.0  90.0  90.0 1   9300.0  12.5
0302F              North Alaska        1  80.0  90.0  90.0 1   9300.0  12.5
0302P              North Alaska        1  80.0  90.0  90.0 1   9300.0  12.5
0303F              North Alaska        1  80.0  90.0  90.0 1   5402.0  12.5
0303P              North Alaska        1  80.0  90.0  90.0 1   5402.0  12.5
0304F              North Alaska        1  80.0  90.0  90.0 2  10980.0  12.5
0304P              North Alaska        1  80.0  90.0  90.0 2  10980.0  12.5
0401F              Pacific Onshore     1  80.0  90.0  90.0 1   7000.0  12.5
0401P              Pacific Onshore     1  80.0  90.0  90.0 1   7000.0  12.5
0402F              Pacific Onshore     1  80.0  90.0  90.0 1   7000.0  12.5
0402P              Pacific Onshore     1  80.0  90.0  90.0 1   7000.0  12.5
0403F              Pacific Onshore     1  80.0  90.0  90.0 1   7000.0  12.5
0403P              Pacific Onshore     1  80.0  90.0  90.0 1   7000.0  12.5
0404F              Pacific Onshore     1  80.0  90.0  90.0 1   7000.0  12.5
0404P              Pacific Onshore     1  80.0  90.0  90.0 1   7000.0  12.5
0407F              Pacific Onshore     1  80.0  90.0  90.0 1   2448.0  12.5
0407P              Pacific Onshore     1  80.0  90.0  90.0 1   2448.0  12.5
0410F              Pacific Onshore     1  80.0  90.0  90.0 1   7000.0  12.5
0410P              Pacific Onshore     1  80.0  90.0  90.0 1   7000.0  12.5
0450F              Pacific Onshore     1  80.0  90.0  90.0 2   2700.0  12.5
0450P              Pacific Onshore     1  80.0  90.0  90.0 2   2700.0  12.5
0451F              Pacific Onshore     1  80.0  90.0  90.0 2   3824.0  12.5
```

.

.

.

(continues with other plays)
Description of File: PLY_DFN.SPC

Data Element	Description	Format
1	Play identifier (4-digit play code)	A20
2	Region name	A20
3	State code (currently not used)	I2
4	First technology (current) development success rate	F6.1
5	Second technology success rate	F6.1
6	Third technology (advanced) development success rate	F6.1
7	Dominant resource type of the play (see below)	I2

8	Average depth of play	F9.1
9	Royalty Rate (%)	F5.1
10	% of play on Federal land	F5.1
11	H_2S Impurity Level	
12	CO_2 Impurity Level	
13	N_2 Impurity Level	
14	Water Depth	
15	Non-Associated USGS Reserves	

Index for dominant resource type (data element 7 above):

NOTE: These are same as the resource types used in the RP Module (resource type indicator of GSAMID). The dominant resource type as used in this file is used to bin reservoirs in specific categories. If a play is predominantly conventional with a few reservoirs being water-drive, then the RP Module uses water-drive type curve to predict the performance of these reservoirs, but in aggregation in the E&P Module all of the reservoirs of the play would be lumped into the conventional category (based on the indicator in Data Element 7 of the PLY_DFN.SPC file, which is the same as the resource type counter of the GSAMID).

The non-associated USGS Reserves are divided into Federal and Private for each play following the procedure explained in the section for the SPEC.DTU file. Their values, though, are not used by the E&P module and serve just for reference purposes. These are used in the Resource Module for splitting the reservoir accumulations into federal and private.

Index	Description
1	Conventional
2	Tight
3	Associated gas
4	Naturally fractured reservoir with induced massive hydraulic fracture
5	Water-drive
6	Unconventional Coal/shale
7	Analyzed resource (currently modeled as Gulf of Mexico offshore reservoirs)

Table C-21

Input Data File: TAX_CDE.SPC (Location: \GSAM\EXPLPROD)
This file contains the tax code and specifications.

```
2000 1
```

Description of File: TAX_CDE.SPC

Data Element	Description	Format
1	year	I4
2	Tax regime (currently modeled as royalty incentive start year)	I2

Table C-22

Input Data File: TAX_DET.SPC (Location: \GSAM\EXPLPROD)
This file contains tax specifications and their detailed definitions. (Not Currently Used).

```
2  0.00  0.00  0.00
```

Description of File: TAX_DET.SPC

Data Element	Description	Format
1	Tax regime	I2
2	Impact on taxes from drilling costs	F6.2
3	Impact on taxes from non-drilling costs	F6.2
4	Impact on the marginal tax rate for changes in the tax code	F6.2

Table C-23

File: UNDBNK.SPC (Location: \GSAM\EXPLPROD)
Contains the developed fraction of undiscovered resource by field size class.

```
Region  Multiple FSC 17 16      15      14      13      12      11      10      9       8       7       6       5       Region Name
1       1.00    0.0000  0.0000  0.0000  0.0500  0.0500  0.0500  0.0500  0.0500  0.0500  0.0500  0.0500  0.0500  0.0500 !Appalachia
2       1.00    0.0000  0.0000  0.0000  0.0000  4.0000  3.0000  2.0000  0.0500  0.0500  0.0500  0.0500  0.0500  0.0500 !MAFLA Onshore
3       1.00    0.0000  0.0000  0.0000  1.0000  1.0000  1.0000  1.0000  1.0000  0.7500  0.7500  0.7500  0.7500  0.7500 !Mid-West
4       1.00    0.0000  0.0000  0.0000  0.0000  0.0000 12.0000  6.0000  1.5000  1.7500  1.7500  1.7500  1.7500  1.7500 !Arkla-East Texas
5       1.00    0.0000  0.0000  0.0000  4.0000  1.0000  1.0000  0.5000  0.3750  0.3750  0.2500  0.2500  1.0000  1.0000 !So-Louisiana
6       1.00    0.0000  0.0000  0.0000  0.0000  0.0000  4.0000  2.0000  0.5000  1.0000  1.0000  0.5000  0.5000  0.5000 !Texas Gulf Coast
7       1.00    0.0000  0.0000  0.0000  0.0000  0.0230  0.0230  0.0058  0.0058  0.0558  2.1057  3.1057  4.1057  4.0057 !Permian
8       1.00    0.0000  0.0000  0.0000  0.0000 16.0000 16.0000  1.2500  1.2500  1.0000  0.5000  0.5000  0.5000  0.5000 !Mid-Continent
9       1.00    0.0000  0.0000  0.0000  0.1235  0.1235  0.3235  0.3000  0.4000  0.5000  0.5000  0.2500  0.2500  0.2500 !San Juan
10      1.10    0.0000  0.0000  0.0000  0.4000  0.0500  0.0500  0.0500  0.0500  0.0375  0.0375  1.0000  1.0000  0.5000 !Rockies Foreland
11      1.00    0.0000  0.0000  0.0000  0.0000  0.0075  0.0075  0.0075  0.0075  0.0075  0.0075  0.0081  0.0088  0.0100 !Williston
12      1.00    0.8563  0.8563  0.8563  1.4272  1.4272  1.5699  1.8553  1.9980  1.9980  1.9980  1.9980  2.1408  2.2835 !Pacific Onshore
13      1.00    0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000 !Atlantic offshore
14      1.00    0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000 !Gulf of Mexico-East
15      1.00    0.0000  0.0000  0.0000  0.1000  0.1000  0.1000  0.1000  0.1000  0.1000  0.1000  0.1000  0.1000  0.1000 !Norphlet
16      1.00    1.0000  1.0000  0.2500  0.2500  0.2500  0.3000  0.3000  0.3000  0.3000  0.3000  0.6000  0.6000  0.6000 !Gulf of Mexico-Cntr
17      1.00    0.2500  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500  0.2500  0.4000  0.4000  0.5000  0.5000  0.5000 !Gulf of Mexico-West
18      1.00    0.0000  0.0000  0.0000  0.0500  0.0500  0.0500  0.0500  0.0500  0.0500  0.0500  0.0500  0.0500  0.0500 !Pacific Offshore
19      1.00    0.5000  1.0000  1.0000  1.0000  1.0000  0.1250  0.1250  0.0625  0.0625  0.0625  0.0625  0.0625  0.0625 !North Alaska
20      1.25    0.5000  1.0000  1.0000  1.0000  1.0000  0.1250  0.1250  0.0625  0.0625  0.0625  0.0625  0.0625  0.0625 !MacKenzie Delta
21      1.25    0.5000  1.0000  1.0000  1.0000  1.0000  0.1250  0.1250  0.0625  0.0625  0.0625  0.0625  0.0625  0.0625 !Not Used
22      1.25    0.5000  1.0000  1.0000  1.0000  1.0000  0.1250  0.1250  0.0625  0.0625  0.0625  0.0625  0.0625  0.0625 !Alberta
23      1.25    0.5000  1.0000  1.0000  1.0000  1.0000  0.1250  0.1250  0.0625  0.0625  0.0625  0.0625  0.0625  0.0625 !British Columbia
24      1.25    0.5000  1.0000  1.0000  1.0000  1.0000  0.1250  0.1250  0.0625  0.0625  0.0625  0.0625  0.0625  0.0625 !Mexico-Supply
0.75    0.85
```

Description of File: UNDBNK.SPC

Data Element	Description	Format
1	Counter	I2
2	Undiscovered resource factor (a factor of 1.1 means the undiscovered resource is 10% more than the Resource Module estimation)	f4.2
3-15	Reserve growth resource factor by Field Size Class (17 to 5)	f6.4
16	GSAM Region Name	format free

Explanation of UNDBNK.SPC

The E&P Module file UNDBNK.SPC controls the distribution of reserve growth resource in the field size classes, as well as a multiplier for the undiscovered resource. The left-hand side number (LHS) in the file is the multiplier. It is a factor that is applied to the undiscovered OGIP in every field size class (actually the NRR from the .GSM file), resulting in a revised estimate of the undiscovered resource in a region (this multiplier is generally kept at unity). The series of numbers to the right of this factor are also multipliers (RHS(FSC)). These numbers are multiplied to the undiscovered amount in a field size class to generate a new number for each field size class. This new number is the total amount available to be assigned to reserve growth (ARG) over the entire GSAM run. Note that these right-hand side factors are on top of the left-hand side multiplier, so that if, in a particular field size class and region, both of the multipliers are 2, then the amount available for reserve growth is *four* times the original undiscovered resource estimate.

The two numbers at the bottom of the file represent the Canadian to U.S. dollar exchange rates. The numbers indicate that the Canadian dollar is 0.75/0.85 = 88% in value compared to the U.S. dollar.

Intended Uses of UNDBNK.SPC

UNDBNK.SPC is used to adjust resource estimates post- Resource and RP Module, and is the only place where reserves growth resource is accounted for. It can be used to test the sensitivity of future market estimates to an increased measure of the resource base, by FSC and region, or by reserves growth or undiscovered resource.

Table C-24

OUTPUT File: DECISION.OUT (Location: \GSAM\EXPLPROD)
This file contains a summary of investment decisions made in the E&P run, used in the Production Accounting Module. This file has been shortened to fit to one page in the Appendix.

```
 13163    392        .3077      1.0769      .6154 2014     0  9  0  0   04114923F006
 13412    392        .3077      1.0769      .6154 2015     0  9  0  0   04114923F006
 13696    392        .0308       .1077      .0615 2016     0  2  0  0   04114923F006
 13697    392        .2769       .9692      .5538 2016     0  9  0  0   04114923F006
 13970    392        .0538       .1885      .1077 2017     0  2  0  0   04114923F006
 13971    392        .1000       .3500      .2000 2017     0  9  0  0   04114923F006
 36739    392        .3077      1.0769      .6154 2011     0  9  0  0   04114923F006
 37848    392        .3077      1.0769      .6154 2012     0  9  0  0   04114923F006
 38987    392        .3077      1.0769      .6154 2013     0  9  0  0   04114923F006
 13164    420        .3784      1.0270      .5946 2014     0  8  0  0   04114926F008
 13413    420        .3784      1.0270      .5946 2015     0  8  0  0   04114926F008
 13698    420        .3784      1.0270      .5946 2016     0  8  0  0   04114926F008
 13972    420        .3784      1.0270      .5946 2017     0  8  0  0   04114926F008
 14255    420        .3784      1.0270      .5946 2018     0  8  0  0   04114926F008
 14585    420        .3784      1.0270      .5946 2019     0  8  0  0   04114926F008
 14888    420        .3784      1.0270      .5946 2020     0  8  0  0   04114926F008
 38988    420        .3784      1.0270      .5946 2013     0  8  0  0   04114926F008
 13414    431        .3478      1.1304      .5217 2015     0  8  0  0   04114927F006
 13699    431        .3478      1.1304      .5217 2016     0  8  0  0   04114927F006
 13973    431        .3478      1.1304      .5217 2017     0  8  0  0   04114927F006
 14256    431        .3478      1.1304      .5217 2018     0  8  0  0   04114927F006
 14586    431        .3478      1.1304      .5217 2019     0  8  0  0   04114927F006
 14889    431        .3478      1.1304      .5217 2020     0  8  0  0   04114927F006
 13165    443        .3077      1.2308      .4615 2014     0 11  0  0   04114929F005
 13415    443        .3077      1.2308      .4615 2015     0 11  0  0   04114929F005
 13700    443        .3077      1.2308      .4615 2016     0 11  0  0   04114929F005
 13974    443        .3077      1.2308      .4615 2017     0 11  0  0   04114929F005
 14257    443        .3077      1.2308      .4615 2018     0 11  0  0   04114929F005
 14587    443        .3077      1.2308      .4615 2019     0 11  0  0   04114929F005
 14890    443        .0154       .0615      .0231 2020     0  4  0  0   04114929F005
 14891    443        .1385       .5538      .2077 2020     0 11  0  0   04114929F005
 13166    444        .3158      1.1579      .5263 2014     0 11  0  0   04114929F006
 13416    444        .3158      1.1579      .5263 2015     0 11  0  0   04114929F006
 13701    444        .3158      1.1579      .5263 2016     0 11  0  0   04114929F006
 13975    444        .3158      1.1579      .5263 2017     0 11  0  0   04114929F006
 14258    444        .3158      1.1579      .5263 2018     0 11  0  0   04114929F006
 14588    444        .3158      1.1579      .5263 2019     0 11  0  0   04114929F006
 14892    444        .0158       .0579      .0263 2020     0  4  0  0   04114929F006
 14893    444        .1421       .5211      .2368 2020     0 11  0  0   04114929F006
 36740    444        .3158      1.1579      .5263 2011     0 11  0  0   04114929F006
 37849    444        .3158      1.1579      .5263 2012     0 11  0  0   04114929F006
```

Description of Output File: DECISION.OUT

Data Element	Description	Format
1	Decision number	I6, 1x
2	Reservoir counter	I6, 1x
3	Wells drilled in pay grade 1	F12.4
4	Wells drilled in pay grade 2	F12.4
5	Wells drilled in pay grade 3	F12.4
6	Start year of primary development program	I5, 1x
7	Start year of secondary development program (“0” means there is none)	I5, 1x
8	Index of technology for primary development (see below)	I2, 1x
9	Index of technology for secondary development (see below)	I2, 1x
10	Secondary development type (see below)	I2, 3x
11	GSAM ID	a12

Index of Development Technology (columns 8 and 9)

Number	Technology
1	Conventional gas reservoir (current)
2	Tight gas reservoir (current)
3	Radial flow reservoirs (current)
4	Linear flow reservoirs (current)
5	Water drive reservoir (current)
6	Coalbed methane (current)
7	Offshore reservoirs (current)
8	Conventional gas reservoir (advanced)
9	Tight gas reservoir (advanced)
10	Radial flow reservoirs (advanced)
11	Linear flow reservoirs (advanced)
12	Water drive reservoir (advanced)
13	Coalbed methane (advanced)
14	Offshore reservoirs (advanced)

Index of Development Technology (column 10)

Number	Technology
0	No secondary development
1	Re-fractured
2	Infill
3	Infill with re-fracture of current well

Table C-25

OUTPUT File: PRICE.OUT (Location: \GSAM\EXPLPROD)
This file contains gas prices, drilling cost factors, number of exploration wells drilled, and their average depths, by year. It is used in the Production Accounting Module. It has been shortened to fit to one page in the Appendix.

```
Pacific Offshore
1993   3.16   .8118   .8118       .0       .0
1994   2.59   .8282   .8037       .0       .0
1995   2.01   .7957   .7957       .0       .0
1996   2.07   .7877   .7877       .0       .0
1997   2.14   .7798   .7798       .0       .0
1998   2.20   .7720   .7720       .0       .0
1999   2.27   .7643   .7643       .0       .0
2000   2.33   .7567   .7567       .0       .0
2001   2.36   .7491   .7491       .0       .0
2002   2.38   .7416   .7416       .0       .0
2003   2.41   .7342   .7342       .0       .0
2004   2.43   .7268   .7268       .0       .0
2005   2.46   .7196   .7196       .0       .0
2006   2.44   .7124   .7124       .0       .0
2007   2.42   .7053   .7053       .0       .0
2008   2.39   .6982   .6982       .0       .0
2009   2.37   .6912   .6912       .0       .0
2010   2.35   .6843   .6843       .0       .0
Pacific Onshore
1993   2.61   .8118  1.0000      9.8   2100.0
1994   2.61   .8661   .9900     11.5   2157.8
1995   2.62   .9801   .9801     13.6   2228.2
1996   2.70   .9703   .9703     15.3   2398.2
1997   2.79   .9606   .9606     16.0   2783.3
1998   2.87   .9510   .9510     18.6   2863.8
1999   2.96   .9415   .9415     16.4   3928.3
2000   3.04   .9321   .9321     18.9   4145.6
2001   3.07   .9227   .9227     20.5   4665.9
2002   3.09   .9135   .9135     31.4   3725.0
2003   3.12   .9044   .9044     31.3   4553.7
2004   3.14   .8953   .8953     41.6   4178.6
2005   3.17   .8864   .8864     48.4   4389.3
2006   3.15   .8775   .8775     61.3   4225.5
2007   3.13   .8687   .8687     69.0   4577.9
2008   3.11   .8601   .8601     82.5   4679.7
2009   3.09   .8515   .8515     98.9   4780.1
2010   3.07   .8429   .8429    121.9   4727.2
San Juan
1993   1.64   .8118  1.0000     29.8   3327.7
1994   1.52   .8037   .9900     17.6   3744.5
1995   1.40   .9801   .9801     18.3   3644.7
1996   1.46   .9703   .9703     17.5   3677.8
1997   1.51   .9606   .9606     16.8   3707.8
1998   1.57   .9510   .9510     16.2   3734.9
1999   1.62   .9415   .9415     15.7   3758.8
2000   1.68   .9321   .9321     20.3   3457.0
```

Description of File: PRICE.OUT

Data Element	Description	Format
1	Year	I4
2	Supply price ($/MCF)	F6.2
3	Development cost factor	F7.4
4	Exploration cost factor	F7.4
5	Number of exploration wells drilled	F8.1
6	Average depth of exploration wells drilled (ft.)	F8.1

Table C-26

OUTPUT File: PRODSUMM.OUT (Location: \GSAM\EXPLPROD)
The file provides the overall supply summary report. The width of the file has been abridged to fit one page.

```
Supply Summary Report
 Region: United States
 Play:   All               Res:
                    1997    1998    1999    2000    2001    2002    2003    2004    2005    2006    2007    2008
 Reserves/Production Summary (BCF):
   BOY Reserves:    306668. 290393. 278390. 266723. 255808. 245297. 236386. 227325. 219013. 211156. 203769. 196726..
   Res. Adds:         2282.   6477.   7581.   9095.  10180.  12135.  12273.  13390.  14325.  15288.  16158.  16894..
   -Production:      18558.  18480.  19248.  20010.  20691.  21046.  21334.  21701.  22182.  22676.  23200.  23792..
   EOY Reserves:    290393. 278390. 266723. 255808. 245297. 236386. 227325. 219013. 211156. 203769. 196726. 189828..

 Drilling Summary (wells):
   Exploration:       1316.   1072.   1226.   1208.   1128.    929.    964.   1007.    970.   1030.   1141.   1286.
   Devl-Primary:       925.   1763.   2880.   3497.   4097.   4787.   5141.   5476.   6087.   6579.   6868.   7169.
   Devl-Infill:          0.    668.    141.     94.    138.    296.     40.     35.     59.     44.     30.     29..
   Total:             2241.   3502.   4247.   4800.   5362.   6012.   6145.   6517.   7116.   7652.   8038.   8484..
   Recompletes:          0.  11035.     16.      0.      0.     27.      4.      0.      0.      0.     16.      0.

   Exp. Success Rate    33.     32.     33.     33.     34.     33.     34.     34.     34.     35.     35.     36..
   Devl Success Rate    80.     80.     80.     80.     80.     80.     80.     80.     80.     80.     80.     80..

 Reserve Addition Summary (Bcf):
   Primary:           2282.   4973.   7284.   8903.  10010.  11403.  12208.  13320.  14229.  15197.  16084.  16827..
   Secondary:            0.   1504.    297.    192.    169.    732.     65.     70.     96.     91.     74.     67.
   Total:             2282.   6477.   7581.   9095.  10180.  12135.  12273.  13390.  14325.  15288.  16158.  16894.

 Reserve Addition Summary (Bcf/Well):
   Primary:            2.47    2.82    2.53    2.55    2.44    2.38    2.37    2.43    2.34    2.31    2.34    2.35
   Secondary:           .00     .13    1.89    2.03    1.23    2.27    1.47    2.02    1.61    2.07    1.61    2.34
   Total:              2.47     .48    2.50    2.53    2.40    2.37    2.37    2.43    2.33    2.31    2.34    2.35

 Production Summary - Pre 1997 Fields:
   Prod (Bcf):       17961.  16860.  16251.  15579.  14860.  13777.  12980.  12094.  11439.  10867.  10369.  10067..
   Number Wells:    160692. 136167. 140198. 134612. 133777. 128275. 124627. 125037. 121967. 117282. 114370. 111248..
Avg Rate (MCA/d/w  306.2   339.2   317.6   317.1   304.3   294.2   285.3   265.0   257.0   253.8   248.4   247.9

 Production Summary - 1997 and Beyond Fields:
   Prod (Bcf):         597.   1620.   2997.   4432.   5831.   7270.   8354.   9607.  10743.  11809.  12831.  13725..
   Number Wells:       832.   2417.   5567.   8810.  12543.  16883.  21494.  25873.  30413.  35032.  39787.  44407..
Avg Rate (Mcf/d/w 1965.6  1836.9  1474.7  1378.0  1273.6  1179.7  1064.8  1017.3   967.7   923.6   883.5   846.8

 Production Summary - All Fields:
   Prod (Bcf):       18558.  18480.  19248.  20010.  20691.  21046.  21334.  21701.  22182.  22676.  23200.  23792.
   Number Wells:    161523. 138584. 145765. 143422. 146320. 145157. 146121. 150910. 152380. 152314. 154157. 155655.
   Avg Rate (Mcf/d/w 314.8   365.3   361.8   382.2   387.4   397.2   400.0   394.0   398.8   407.9   412.3   418.8
   R/P ratio:         16.65   16.06   14.86   13.78   12.86   12.23   11.66   11.09   10.52    9.99    9.48    8.98

    WH Price ($/Mca)   2.19    1.90    2.05    2.11    2.21    2.26    2.27    2.26    2.25    2.26    2.28    2.31
    Revenues ($MM):  39574.  34668.  38699.  40408.  42923.  44730.  46450.  48112.  50408.  52906.  54897.  56822.

 Original Gas in Place Summary (Tcf):

 Undiscovered Resource (OGIP) at BOY
  Sz Class:  5       89.634  88.713  87.885  86.855  85.780  84.729  83.840  82.866  81.801  80.778  79.636  78.363
  Sz Class:  6       94.234  92.927  91.733  90.336  88.907  87.519  86.288  84.996  83.541  82.123  80.552  78.770
  Sz Class:  7       97.765  95.680  93.929  91.828  89.815  87.863  86.048  84.144  81.963  79.863  77.500  74.869
  Sz Class:  8       99.358  96.445  94.142  91.419  88.875  86.279  83.881  81.371  78.488  75.756  72.816  69.624
  Sz Class:  9      101.399  97.184  93.821  90.263  86.731  83.222  79.986  76.724  73.277  69.812  65.909  61.646
  Sz Class: 10      106.014  99.774  94.142  88.298  82.443  76.382  71.063  65.772  60.623  55.983  51.588  47.606
  Sz Class: 11      108.860  99.692  90.926  82.004  73.922  66.750  61.072  56.192  52.544  49.211  45.085  41.625
  Sz Class: 12      113.709  95.593  82.169  71.580  63.968  58.077  55.113  52.244  49.773  46.723  43.122  39.685
  Sz Class: 13      105.129  86.946  74.302  67.041  62.257  57.392  55.057  52.894  49.705  46.215  44.673  42.980
  Sz Class: 14      110.625  91.516  80.419  74.711  69.914  64.205  62.195  59.649  52.769  49.975  48.195  46.121
  Sz Class: 15       70.426  66.330  62.911  60.007  57.514  53.946  52.430  51.027  49.727  48.521  47.070  45.438
  Sz Class: 16       27.814  27.288  26.782  26.296  25.829  25.379  24.948  24.533  24.134  23.750  23.211  22.543
  Sz Class: 17         .000    .000    .000    .000    .000    .000    .000    .000    .000    .000    .000    .000
  Total:            1124.97 1038.09  973.16  920.64  875.95  831.74  801.92  772.41  738.34  708.71  679.36  649.27

 Banked Resource (OGIP) at BOY
  Sz Class:  5        9.425  14.443  15.684  17.073  18.165  19.229  20.122  21.097  22.136  23.058  24.072  25.177
  Sz Class:  6        9.384  14.337  15.687  17.168  18.354  19.477  20.416  21.358  22.469  23.423  24.508  25.743
  Sz Class:  7        8.089  13.155  14.862  16.817  18.322  19.712  20.859  21.952  23.243  24.318  25.641  27.122
  Sz Class:  8        7.162  12.948  15.027  17.457  19.378  21.306  22.866  24.395  26.206  27.749  29.433  31.214
  Sz Class:  9        6.197  12.718  15.734  18.833  21.597  24.174  26.387  28.443  30.559  32.557  34.852  37.350
  Sz Class: 10        5.053  13.390  18.719  23.977  28.818  33.547  37.296  40.839  44.089  46.611  48.740  50.234
  Sz Class: 11       10.782  24.598  32.985  41.002  47.545  52.811  56.279  58.700  59.611  60.135  61.275  61.585
  Sz Class: 12       35.957  69.030  81.620  90.521  94.861  97.207  96.330  95.213  93.518  91.939  90.851  89.456
  Sz Class: 13        4.234  23.916  36.137  42.366  45.577  48.604  48.968  49.076  50.164  51.416  50.359  49.160
  Sz Class: 14        3.492  24.329  35.379  40.387  44.121  48.704  49.243  50.286  55.262  55.774  54.845  54.176
  Sz Class: 15        3.342   9.556  13.188  16.303  18.881  22.533  23.880  25.089  25.827  26.482  27.380  28.458
  Sz Class: 16        1.358   1.884   2.390   2.876   3.343   3.792   4.224   4.639   5.038   5.422   5.961   6.629
  Sz Class: 17         .000    .000    .000    .000    .000    .000    .000    .000    .000    .000    .000    .000
  Total:             104.48  234.30  297.41  344.78  378.96  411.10  426.87  441.09  458.12  468.88  477.91  486.30

 Reserve Growth Resource (OGIP) at BOY
  Sz Class:  5       44.195  39.874  39.303  38.732  38.490  38.249  38.007  37.765  37.523  37.327  37.131  36.934
  Sz Class:  6       42.221  38.217  37.679  37.142  36.911  36.680  36.449  36.218  35.987  35.815  35.643  35.471
  Sz Class:  7       37.559  34.170  33.729  33.289  33.098  32.907  32.715  32.524  32.333  32.187  32.041  31.896
  Sz Class:  8       31.062  27.877  27.488  27.099  26.926  26.753  26.579  26.406  26.233  26.098  25.964  25.829
  Sz Class:  9       24.299  21.734  21.448  21.162  21.025  20.888  20.750  20.613  20.476  20.381  20.285  20.190
  Sz Class: 10       20.073  17.972  17.731  17.490  17.377  17.264  17.151  17.038  16.924  16.844  16.763  16.682
  Sz Class: 11       35.184  30.341  29.915  29.488  29.254  29.020  28.787  28.553  28.319  28.158  27.996  27.834
  Sz Class: 12      125.634 109.227 107.225 105.223 104.299 103.375 102.450 101.526 100.602  99.980  99.358  98.736
  Sz Class: 13       18.481  16.897  16.738  16.580  16.500  16.421  16.342  16.263  16.183  16.124  16.066  16.007
  Sz Class: 14       24.404  22.676  22.481  22.285  22.209  22.132  22.055  21.979  21.902  21.801  21.700  21.599
  Sz Class: 15       26.352  24.233  24.021  23.810  23.725  23.640  23.555  23.471  23.386  23.259  23.132  23.004
  Sz Class: 16       12.218  12.218  12.218  12.218  12.218  12.218  12.218  12.218  12.218  12.218  12.218  12.218
  Sz Class: 17         .000    .000    .000    .000    .000    .000    .000    .000    .000    .000    .000    .000    .
  Total:             441.68  395.44  389.98  384.52  382.03  379.55  377.06  374.57  372.09  370.19  368.30  366.40
```

Table C-27

OUTPUT File: REVSUMM.OUT (Location: \GSAM\EXPLPROD)
The file provides a summary of the different types of reserves by region.

```
                                                Reserve Summary Report
Region: United States
Type:   All

Total Undiscovered and Undeveloped Resource (TCF  1402.150
Total Undiscovered Resource (TCF):                1243.267
Discovered Undeveloped Resource (TCF):             158.883
Total Discovered (TCF):                            491.469
Total Available for Development (TCF):             650.353
Total Developed (TCF):                             461.649



Initial Reserves (TCF):                            248.361
Reserve Additions (Prim.) (TCF):                   313.657
Reserve Additions (Sec.) (TCF):                     23.847
Total Reserves/Prod (TCF):                         585.866

                                                Reserve Summary Report
Region: Canada
Type:   All

Total Undiscovered and Undeveloped Resource (TCF   743.274
Total Undiscovered Resource (TCF):                 583.661
Discovered Undeveloped Resource (TCF):             159.612
Total Discovered (TCF):                            219.264
Total Available for Development (TCF):             378.876
Total Developed (TCF):                             283.156



Initial Reserves (TCF):                             49.659
Reserve Additions (Prim.) (TCF):                   105.514
Reserve Additions (Sec.) (TCF):                      1.730
Total Reserves/Prod (TCF):                         156.903

                                                Reserve Summary Report
Region: San Juan
Type:   All

Total Undiscovered and Undeveloped Resource (TCF    69.146
Total Undiscovered Resource (TCF):                  67.844
Discovered Undeveloped Resource (TCF):               1.302
Total Discovered (TCF):                             45.998
Total Available for Development (TCF):              47.300
Total Developed (TCF):                              29.520
```

Table C-28

OUTPUT File: SUPPSUMM.OUT (Location: \GSAM\EXPLPROD)
This file contains supply summary by region and resource type. It has been abridged to fit the width of the page.

```
                          1997      1998      1999      2000      2001      2002      2003
Pacific Offshore          49.4      49.4      49.4      53.3      43.5      62.1      60.0
Pacific Onshore          195.2     198.5     215.6     241.2     267.5     316.0     342.3
San Juan                1255.9    1246.5    1251.6    1255.5    1234.8    1107.7    1048.0
Rockies Foreland        1733.0    1805.0    1987.4    2233.8    2484.7    2791.2    3091.2
Williston                 74.1      71.4      79.8      92.1     103.7     136.9     164.6
Permian                 1611.0    1591.3    1605.8    1649.2    1619.7    1663.0    1692.4
Mid-Continent           2521.4    2526.8    2660.9    2803.6    2919.5    3024.7    3047.8
Arkla-East Texas        1464.9    1538.9    1609.0    1690.9    1809.1    1875.3    1937.6
Texas Gulf Coast        2319.8    2167.3    2208.3    2318.8    2347.7    2392.2    2372.8
Gulf of Mexico-West     1334.1    1422.9    1578.7    1738.6    1804.1    1821.1    1818.2
Gulf of Mexico-Cntr     3538.0    3644.9    3676.8    3592.6    3586.5    3334.3    3154.1
Norphlet                 285.6     253.7     241.4     167.2     160.2     157.4     155.2
Gulf of Mexico-East         .0        .0        .0        .0        .0        .0        .0
So-Louisiana             950.5     856.2     871.3     873.0     885.2     858.6     833.9
MAFLA Onshore            479.2     466.4     480.7     504.5     517.0     507.1     519.0
Mid-West                  79.2      87.0     106.3     118.9     153.5     169.0     195.3
Appalachia               666.3     553.8     624.8     677.2     754.0     829.7     901.6
Alberta                 4969.2    4574.0    4419.6    4364.9    4434.6    4495.3    4488.2
British Columbia         640.2     614.1     630.4     657.1     693.0     745.9     761.8
North Alaska                .0        .0        .0        .0        .0        .0        .0
MacKenzie Delta             .0        .0        .0        .0        .0        .0        .0
Atlantic Offshore           .0        .0        .0        .0        .0        .0        .0
Mexico-Supply             70.0      72.0      75.0      77.0      80.0      80.0      85.0

Total U.S.             18557.7   18480.3   19247.8   20010.3   20690.8   21046.3   21334.0
Conventional            6851.8    6647.0    6926.0    7223.9    7520.8    7731.7    7834.5
Tight                   2770.0    2762.5    3069.3    3440.3    3848.3    4349.5    4810.6
Associate-Dissolved     2655.7    2655.7    2655.7    2708.7    2757.7    2573.1    2535.3
Natural Fracture         232.7     224.3     218.1     215.5     216.0     218.5     220.5
Water Drive              814.6     741.6     703.4     675.5     630.5     597.0     563.2
Coal and Shale          1122.3    1138.7    1175.4    1207.2    1228.7    1112.8    1083.2
Analyzed Resource       4110.6    4310.5    4499.9    4539.2    4488.7    4463.7    4286.7

Total Canada            5609.4    5188.2    5050.0    5022.1    5127.5    5241.2    5250.0
Conventional            2899.1    2651.6    2510.9    2454.6    2552.4    2650.7    2675.2
Tight                      5.1       5.8       6.3       6.5       7.9       9.1      10.6
Associate-Dissolved     1041.0    1027.0    1027.0    1027.0    1011.0     995.0     979.0
Natural Fracture        1634.3    1442.1    1389.6    1360.9    1319.1    1272.1    1176.3
Water Drive                7.3       7.3       7.3       7.3       7.3      13.6      13.2
Coal and Shale            22.6      54.5     108.9     165.7     229.8     300.6     395.7
Analyzed Resource                 .0        .0        .0        .0        .0        .0        .0
```

Table C-29

OUTPUT File: SUPPLY.EXT (Location: \GSAM\EXPLPROD)

This file contains summary data for the wellhead gas price and the supply of natural gas by year and region. It has been abridged to fit the width of the page.

```
                         1993   1993  1994   1994  1995   1995  1996   1996  1997   1997
Pacific Offshore      1  1.92   11.0  1.84   10.0  1.75    8.0  1.71    9.0  1.75   49.4
Pacific Onshore       1  1.92  101.0  1.84   92.0  1.75   92.0  1.70   76.0  1.75  195.2
San Juan              1  1.74  861.0  1.66  879.0  1.57  953.0  1.52  975.0  1.74 1255.9
Rockies Foreland      1  1.73 1067.0  1.61 1221.0  1.49 1261.0  1.52 1311.0  1.74 1733.0
Williston             1  1.65   59.0  1.54   62.0  1.42   58.0  1.45   62.0  1.74   74.1
Permian               1  1.81  986.0  1.72 1017.0  1.63  902.0  1.59  917.0  2.06 1611.0
Mid-Continent         1  1.74 2840.0  1.66 2860.0  1.58 2697.0  1.53 2774.0  2.19 2521.4
Arkla-East Texas      1  1.85 1165.0  1.76 1164.0  1.67 1184.0  1.62 1240.0  2.19 1464.9
Texas Gulf Coast      1  1.78 1855.0  1.70 1927.0  1.61 2024.0  1.56 2155.0  2.19 2319.8
Gulf of Mexico-West   1  1.78 1206.0  1.69 1206.0  1.61 1146.0  1.56 1172.0  2.26 1334.1
Gulf of Mexico-Cntr   1  1.82 2841.0  1.74 2934.0  1.66 2847.0  1.61 3167.0  2.26 3538.0
Norphlet              1  1.84  285.6  1.76  253.7  1.67  241.4  1.63  167.2  2.26  285.6
Gulf of Mexico-East   1   .00     .0   .00     .0   .00     .0   .00     .0   .00     .0
So-Louisiana          1  1.85  895.0  1.76  883.0  1.68  838.0  1.64  879.0  2.19  950.5
MAFLA Onshore         1  1.80  352.0  1.71  435.0  1.63  417.0  1.59  426.0  2.19  479.2
Mid-West              1  1.87  105.0  1.79  109.0  1.70  124.0  1.65  170.0  2.38   79.2
Appalachia            1  1.96  466.0  1.88  500.0  1.80  455.0  1.76  471.0  2.56  666.3
Alberta               1  1.23 4162.0  1.29 4505.0  1.44 4663.0  1.44 4900.0  2.19 4969.2
British Columbia      1  1.59  580.0  1.49  616.0  1.40  681.0  1.50  685.0  2.19  640.2
North Alaska          1   .00     .0   .00     .0   .00     .0   .00     .0   .00     .0
MacKenzie Delta       1   .00     .0   .00     .0   .00     .0   .00     .0   .00     .0
Atlantic Offshore     1   .00     .0   .00     .0   .00     .0   .00     .0   .00     .0
Mexico-Supply         1  1.80   70.0  1.72   72.0  1.64   75.0  1.53   77.0  1.21   70.0
```

Description:

```
Data Element 1: GSAM Region Name
Data Element 2: Not used currently
Data Element 3: Price track
Data Element 4: Wellhead gas price ($/MCF) in 1993
Data Element 5: Supply in BCF in 1993
Data Element 6: Well head gas price ($/MCF) in 1994
Data Element 7: Supply in BCF in 1994
Data Element 8: Well head gas price ($/MCF) in 1995
Data Element 9: Supply in BCF in 1995
Data Element 10:        Well head gas price ($/MCF) in 1996
Data Element 11:        Supply in BCF in 1996
Data Element 12:        Well head gas price ($/MCF) in 1997
Data Element 13:        Supply in BCF in 1997
```

Table C-30

OUTPUT File: UNRRSUMM.OUT (Location: \GSAM\EXPLPROD)

This file contains summary data for the Undiscovered Recoverable Reserves by year, FSC, and region. It has been abridged to fit the width of the page.

```
Rec. Reserve Summary Report
 Region: United States
 Play:   All                 Res:
                     1997    1998    1999    2000    2001    2002    2003

 Undisc. Curr. Tech. Rec. Resv. at BOY (TCF)
  Sz Class:  5       56.987  56.357  55.787  55.076  54.330  53.602  52.992
  Sz Class:  6       59.265  58.372  57.553  56.598  55.622  54.679  53.849
  Sz Class:  7       62.488  61.062  59.864  58.431  57.061  55.733  54.511
  Sz Class:  8       62.727  60.748  59.197  57.361  55.666  53.949  52.378
  Sz Class:  9       64.552  61.675  59.412  57.040  54.697  52.363  50.253
  Sz Class: 10       70.293  66.028  62.196  58.274  54.343  50.296  46.797
  Sz Class: 11       72.295  66.030  60.049  54.067  48.755  44.026  40.255
  Sz Class: 12       76.273  63.900  54.795  47.663  42.516  38.536  36.643
  Sz Class: 13       69.873  57.403  48.981  44.116  40.929  37.662  36.221
  Sz Class: 14       73.670  60.113  52.418  48.559  45.352  41.363  40.149
  Sz Class: 15       42.977  40.246  38.016  36.163  34.603  32.141  31.307
  Sz Class: 16       16.817  16.531  16.256  15.991  15.737  15.492  15.257
  Sz Class: 17         .000    .000    .000    .000    .000    .000    .000
  Total:             728.22  668.47  624.52  589.34  559.61  529.84  510.61

 Undisc. Adv. Tech. Rec. Resv. at BOY (TCF)
  Sz Class:  5       61.120  60.453  59.854  59.109  58.329  57.568  56.932
  Sz Class:  6       64.263  63.318  62.460  61.456  60.432  59.443  58.575
  Sz Class:  7       67.849  66.343  65.085  63.580  62.142  60.750  59.472
  Sz Class:  8       68.330  66.239  64.610  62.677  60.890  59.080  57.433
  Sz Class:  9       70.840  67.796  65.417  62.918  60.446  57.990  55.773
  Sz Class: 10       75.459  70.962  66.954  62.833  58.706  54.457  50.792
  Sz Class: 11       77.108  70.536  64.313  58.061  52.471  47.513  43.602
  Sz Class: 12       80.573  67.615  58.148  50.727  45.358  41.168  39.178
  Sz Class: 13       73.576  60.615  51.824  46.778  43.433  39.995  38.442
  Sz Class: 14       76.934  63.117  55.220  51.199  47.855  43.641  42.349
  Sz Class: 15       45.038  42.169  39.821  37.864  36.213  33.638  32.739
  Sz Class: 16       17.428  17.119  16.822  16.537  16.262  15.998  15.745
  Sz Class: 17         .000    .000    .000    .000    .000    .000    .000
  Total:             778.52  716.28  670.53  633.74  602.54  571.24  551.03
```

.

.

.

(continues down with other regions)

Table C-31

OUTPUT File: BNRRSUMM.OUT (Location: \GSAM\EXPLPROD)

This file contains summary data for the Banked Recoverable Reserves by year (1997-2020), FSC, and region. It has been abridged to fit the width of the page. The excerpt shows the data for the United States.

```
Rec. Reserve Summary Report
 Region: United States
 Play:   All                 Res:
                     1997    1998    1999    2000    2001    2002    2003

 Banked Curr. Tech. Rec. Resv. at BOY (TCF)
  Sz Class:  5        6.268   9.628  10.474  11.421  12.178  12.912  13.526
  Sz Class:  6        6.190   9.532  10.462  11.477  12.298  13.067  13.703
  Sz Class:  7        5.311   8.733   9.907  11.244  12.287  13.240  14.016
  Sz Class:  8        4.713   8.650  10.083  11.752  13.056  14.349  15.384
  Sz Class:  9        4.031   8.467  10.536  12.633  14.496  16.235  17.687
  Sz Class: 10        3.382   9.082  12.750  16.326  19.635  22.842  25.345
  Sz Class: 11        7.247  16.669  22.429  27.846  32.182  35.705  38.081
  Sz Class: 12       24.919  47.691  56.281  62.323  65.283  66.893  66.215
  Sz Class: 13        3.071  16.677  24.791  28.978  31.085  33.116  33.252
  Sz Class: 14        2.292  17.024  24.687  28.036  30.484  33.672  33.830
  Sz Class: 15        2.117   6.246   8.616  10.609  12.224  14.743  15.427
  Sz Class: 16         .739   1.025   1.301   1.565   1.820   2.064   2.299
  Sz Class: 17         .000    .000    .000    .000    .000    .000    .000
  Total:              70.28  159.42  202.32  234.21  257.03  278.84  288.76

 Banked Adv. Tech. Rec. Resv. at BOY (TCF)
  Sz Class:  5        6.591  10.117  11.004  11.999  12.790  13.558  14.196
  Sz Class:  6        6.535  10.045  11.022  12.092  12.952  13.760  14.427
  Sz Class:  7        5.657   9.273  10.510  11.920  13.012  14.014  14.829
  Sz Class:  8        5.005   9.149  10.655  12.414  13.790  15.156  16.241
  Sz Class:  9        4.307   8.993  11.167  13.377  15.341  17.171  18.698
  Sz Class: 10        3.604   9.612  13.446  17.200  20.661  24.015  26.625
  Sz Class: 11        7.623  17.500  23.489  29.147  33.712  37.396  39.827
  Sz Class: 12       26.069  49.895  58.821  65.097  68.179  69.888  69.185
  Sz Class: 13        3.232  17.380  25.867  30.207  32.436  34.584  34.772
  Sz Class: 14        2.396  17.412  25.281  28.793  31.377  34.779  34.992
  Sz Class: 15        2.165   6.444   8.933  11.031  12.738  15.369  16.114
  Sz Class: 16         .797   1.106   1.403   1.689   1.963   2.227   2.481
  Sz Class: 17         .000    .000    .000    .000    .000    .000    .000
  Total:              73.98  166.93  211.60  244.96  268.95  291.92  302.39
.
.
.
```

(continues down with other regions)

Table C-32

OUTPUT File: RGRRSUMM.OUT (Location: \GSAM\EXPLPROD)

This file contains summary data for the Reserve Growth of Recoverable Reserves by year, FSC, and region. It has been abridged to fit the width of the page. The excerpt shows the data for the United States.

```
Rec. Reserve Summary Report
 Region: United States
 Play:   All                 Res:
                     1997    1998    1999    2000    2001    2002    2003

 Resv. Grw. Curr. Tech. Rec. Resv. at BOY (TCF)
  Sz Class:  5       29.546  26.661  26.280  25.899  25.737  25.576  25.414
  Sz Class:  6       27.785  25.104  24.744  24.384  24.230  24.075  23.920
  Sz Class:  7       24.575  22.313  22.018  21.724  21.597  21.469  21.341
  Sz Class:  8       20.228  18.082  17.819  17.556  17.439  17.322  17.206
  Sz Class:  9       15.567  13.854  13.663  13.471  13.379  13.288  13.196
  Sz Class: 10       13.150  11.714  11.546  11.379  11.302  11.224  11.146
  Sz Class: 11       23.219  19.933  19.642  19.350  19.191  19.032  18.873
  Sz Class: 12       86.673  75.272  73.873  72.474  71.830  71.186  70.542
  Sz Class: 13       12.313  11.107  10.988  10.869  10.809  10.749  10.688
  Sz Class: 14       15.497  14.321  14.187  14.052  14.000  13.947  13.895
  Sz Class: 15       16.536  15.138  14.998  14.858  14.802  14.746  14.691
  Sz Class: 16        6.650   6.650   6.650   6.650   6.650   6.650   6.650
  Sz Class: 17         .000    .000    .000    .000    .000    .000    .000
  Total:             291.74  260.15  256.41  252.67  250.97  249.26  247.56

 Resv. Grw. Adv. Tech. Rec. Resv. at BOY (TCF)
  Sz Class:  5       30.944  27.926  27.526  27.125  26.956  26.786  26.616
  Sz Class:  6       29.200  26.390  26.012  25.633  25.470  25.307  25.145
  Sz Class:  7       26.005  23.616  23.303  22.990  22.855  22.719  22.584
  Sz Class:  8       21.356  19.102  18.824  18.546  18.423  18.300  18.176
  Sz Class:  9       16.523  14.718  14.513  14.309  14.212  14.114  14.017
  Sz Class: 10       13.947  12.433  12.256  12.078  11.995  11.913  11.830
  Sz Class: 11       24.438  20.999  20.692  20.385  20.219  20.052  19.885
  Sz Class: 12       90.749  78.835  77.371  75.907  75.233  74.559  73.886
  Sz Class: 13       13.048  11.789  11.663  11.537  11.474  11.410  11.347
  Sz Class: 14       16.098  14.898  14.759  14.621  14.567  14.512  14.458
  Sz Class: 15       16.971  15.561  15.420  15.279  15.222  15.166  15.109
  Sz Class: 16        7.175   7.175   7.175   7.175   7.175   7.175   7.175
  Sz Class: 17         .000    .000    .000    .000    .000    .000    .000
  Total:             306.45  273.44  269.51  265.58  263.80  262.01  260.23
.
.
.
```
(continues down with other regions)

Table C-33

OUTPUT File: NRRSUMM.OUT (Location: \GSAM\EXPLPROD)

This file contains summary data for the Number of Reservoir Accumulations by year, FSC, and region. It has been abridged to fit the width of the page. The excerpt shows the data for the United States.

```
NRR Summary Report
 Region: United States
 Play:   All                 Res:
                     1997    1998    1999    2000    2001    2002    2003

 NRR Summary (rounded to nearest integer):
  Sz Class:  5         9987    9871    9775    9655    9530    9412    9319
  Sz Class:  6         5505    5420    5349    5264    5177    5095    5028
  Sz Class:  7         2985    2915    2861    2796    2733    2673    2621
  Sz Class:  8         1575    1526    1490    1446    1404    1363    1328
  Sz Class:  9          832     793     765     735     705     676     651
  Sz Class: 10          439     412     389     365     342     318     299
  Sz Class: 11          228     208     191     174     159     144     134
  Sz Class: 12          120     102      90      80      72      65      62
  Sz Class: 13           59      50      44      40      37      34      32
  Sz Class: 14           31      26      23      22      20      19      18
  Sz Class: 15           10       9       9       8       8       8       7
  Sz Class: 16            2       2       2       2       2       2       2
  Sz Class: 17            0       0       0       0       0       0       0
  Total:              21773   21335   20988   20587   20189   19809   19502
.
.
.
```

(continues down with other regions)

Table C-34

INPUT File: RESAV.SPC (Location: \GSAM\EXPLPROD)

The file has been abridged to fit the length of the page

```
Conv
Tight
Rad Flow
Lin Flow
W Drive
Unconv
Analyzed
All Regions    Pri%  Fed%
Conv     1997   0.0   0.0
Conv     1998  30.0  30.0
Conv     2000  30.0  30.0
Conv     2005  30.0  30.0
Conv     2010  40.0  40.0
Conv     2015  50.0  50.0
Conv     2020  70.0  70.0
Tight    1997   0.0   0.0
Tight    1998  20.0  20.0
Tight    2000  20.0  20.0
Tight    2005  20.0  20.0
Tight    2010  25.0  25.0
Tight    2015  35.0  35.0
Tight    2020  70.0  70.0
Rad Flow 1997   0.0   0.0
Rad Flow 1998  20.0  20.0
Rad Flow 2000  20.0  20.0
Rad Flow 2005  20.0  20.0
Rad Flow 2010  25.0  25.0
Rad Flow 2015  35.0  35.0
Rad Flow 2020  70.0  70.0
Lin Flow 1997   0.0   0.0
Lin Flow 1998  20.0  20.0
Lin Flow 2000  20.0  20.0
Lin Flow 2005  20.0  20.0
Lin Flow 2010  25.0  25.0
Lin Flow 2015  35.0  35.0
Lin Flow 2020  70.0  70.0
W Drive  1997   0.0   0.0
W Drive  1998  20.0  20.0
W Drive  2000  20.0  20.0
W Drive  2005  20.0  20.0
W Drive  2010  25.0  25.0
W Drive  2015  35.0  35.0
W Drive  2020  70.0  70.0
.
.
.
```
(continues down with other regions)

Explanation of RESAV.SPC

As an integral part of Federal/Private land modeling, RESAV.SPC was added to the E&P module to control discovered/undeveloped reserve availability. The file stores regional reserve availability percentage of each resource type for Federal and Private lands as a function of time.

Descriptions:

```
Header line             Region Name
Data Element 1: Resource Type
Data Element 2: Year
Data Element 3: Federal Land Reserve Availability (%)
Data Element 4: Private Land Reserve Availability (%)
Index for end “End all-1”
of regional data
(not shown in the excerpt)
```

Table C-35

INPUT File: RESAVRG.SPC (Location: \GSAM\EXPLPROD)

```
Conv
Tight
Rad Flow
Lin Flow
W Drive
Unconv
```

```
Analyzed
All Regions !
Conv     1997   0.0
Tight    1997   0.0
Rad Flow 1997   0.0
Lin Flow 1997   0.0
W Drive  1997   0.0
Unconv   1997   0.0
Analyzed 1997   0.0
Conv     1998  29.0
Tight    1998  29.0
Rad Flow 1998  29.0
Lin Flow 1998  29.0
W Drive  1998  29.0
Unconv   1998  29.0
Analyzed 1998  29.0
Conv     2000  32.0
Tight    2000  32.0
Rad Flow 2000  32.0
Lin Flow 2000  32.0
W Drive  2000  32.0
Unconv   2000  32.0
Analyzed 2000  32.0
Conv     2005  35.0
Tight    2005  35.0
Rad Flow 2005  35.0
Lin Flow 2005  35.0
W Drive  2005  35.0
Unconv   2005  35.0
Analyzed 2005  35.0
Conv     2010  37.0
Tight    2010  37.0
Rad Flow 2010  37.0
Lin Flow 2010  37.0
W Drive  2010  37.0
Unconv   2010  37.0
Analyzed 2010  37.0

.
.
.
```
(continues down with other regions)

<u>Descriptions:</u>

```
Header line            Region Name
Data Element 1: Resource Type
Data Element 2: Year
Data Element 3: Reserve Growth Rate (%)
Index for end “End all-1”
of regional data
(not shown in the excerpt)
```

Explanation of RESAVRG.SPC

This file contains reserve growth rate by resource type, region, and year. This value controls the reserve growth resource availability by year. The current values in the file reflect the exact projections of the USGS for the regional reserve growth rate through the year 2020.

Table C-36

INPUT File: ETEC_FED.SPC (Location: \GSAM\EXPLPROD)
This File contains Federal Lands Technology Penetration Increments for Exploration Technology
NOTE: Values Should be Specified for all the years.

```
Federal Lands Technology Penetration Increments For EXPLORATION TECHNOLOGY
NOTE: Values Should be Specified for all the years
Year  Curr. Tech. Increment     Adv. Tech. Increment
1997        0                        0
1998        0                        0
1999        0                        0
2000        0                        0
2001        0                        0
2002        0                        0
2003        0                        0
```

```
2004        0                        0
2005        0                        0
2006        0                        0
2007        0                        0
2008        0                        0
2009        0                        0
2010        0                        0
2011        0                        0
2012        0                        0
2013        0                        0
2014        0                        0
2015        0                        0
2016        0                        0
2017        0                        0
2019        0                        0
2020        0                        0
```

Description:

Data Element 1:	Year
Data Element 2:	Current Technology Increment
Data Element 3:	Advancement Technology Increment

Explanation of ETEC_FED.SPC

The file ETEC_FED.SPC stores current and advanced technology incremental penetration rates as a function of time for exploration drilling program.

Table C-37

INPUT File: DTEC_FED.SPC (Location: \GSAM\EXPLPROD)
This File contains Federal Lands Technology Penetration Increments for Development Technology
NOTE: Values Should be Specified for all the years.

```
Federal Lands Technology Penetration Increments For DEVELOPMENT TECHNOLOGY
NOTE: Values Should be Specified for all the years
Year  Curr. Tech. Increment     Adv. Tech. Increment
1997        0                        0
1998        0                        0
1999        0                        0
2000        0                        0
2001        0                        0
2002        0                        0
2003        0                        0
2004        0                        0
2005        0                        0
```

2006	0	0
2007	0	0
2008	0	0
2009	0	0
2010	0	0
2011	0	0
2012	0	0
2013	0	0
2014	0	0
2015	0	0
2016	0	0
2017	0	0
2019	0	0
2020	0	0

Description:

Data Element 1:	Year
Data Element 2:	Current Technology Increment
Data Element 3:	Advancement Technology Increment

Explanation of DTEC_FED.SPC

The file DTEC_FED.SPC stores current and advanced technology incremental penetration rates as a function of time for development drilling program.

Table C-38

INPUT File: SUPPLY.HIS (Location: \GSAM\EXPLPROD)
This file contains historical data for the wellhead gas price and the supply of natural gas by year (1993-1996) and GSAM region. It has been abridged to fit the width of the page. This file is not used directly in the Exploration and Production Module but rather serves to transfer the historical data to the Demand Integrating Module.

		1993	1993	1994	1994	1995	1995	1996	1996
Pacific Offshore	1	1.92	11.0	1.84	10.0	1.75	8.0	1.71	9.0
Pacific Onshore	1	1.92	101.0	1.84	92.0	1.75	92.0	1.70	76.0
San Juan	1	1.74	861.0	1.66	879.0	1.57	953.0	1.52	975.0
Rockies Foreland	1	1.73	1067.0	1.61	1221.0	1.49	1261.0	1.52	1311.0
Williston	1	1.65	59.0	1.54	62.0	1.42	58.0	1.45	62.0
Permian	1	1.81	986.0	1.72	1017.0	1.63	902.0	1.59	917.0
Mid-Continent	1	1.74	2840.0	1.66	2860.0	1.58	2697.0	1.53	2774.0
Arkla-East Texas	1	1.85	1165.0	1.76	1164.0	1.67	1184.0	1.62	1240.0
Texas Gulf Coast	1	1.78	1855.0	1.70	1927.0	1.61	2024.0	1.56	2155.0
Gulf of Mexico-West	1	1.78	1206.0	1.69	1206.0	1.61	1146.0	1.56	1172.0
Gulf of Mexico-Cntr	1	1.82	2841.0	1.74	2934.0	1.66	2847.0	1.61	3167.0
Norphlet	1	1.84	285.6	1.76	253.7	1.67	241.4	1.63	167.2
Gulf of Mexico-East	1	0.00	.0	0.00	.0	0.00	.0	0.00	.0
So-Louisiana	1	1.85	895.0	1.76	883.0	1.68	838.0	1.64	879.0

```
MAFLA Onshore        1  1.80  352.0  1.71  435.0  1.63  417.0  1.59  426.0
Mid-West             1  1.87  105.0  1.79  109.0  1.70  124.0  1.65  170.0
Appalachia           1  1.96  466.0  1.88  500.0  1.80  455.0  1.76  471.0
Alberta              1  1.19 4162.0  1.32 4505.0  0.94 4663.0  1.09 4900.0
British Columbia     1  0.99  580.0  1.04  616.0  0.76  681.0  0.82  685.0
North Alaska         1  0.00     .0  0.00     .0  0.00     .0  0.00     .0
MacKenzie Delta      1  0.00     .0  0.00     .0  0.00     .0  0.00     .0
Atlantic Offshore    1  0.00     .0  0.00     .0  0.00     .0  0.00     .0
Mexico-Supply        1  1.80   70.0  1.72   72.0  1.64   75.0  1.53   77.0
```

Description:

```
Data Element 1: GSAM Region Name
Data Element 2: Price track
Data Element 3: Wellhead gas price ($/MCF) in 1993
Data Element 4: Supply in BCF in 1993
Data Element 5: Well head gas price ($/MCF) in 1994
Data Element 6: Supply in BCF in 1994
Data Element 7: Well head gas price ($/MCF) in 1995
Data Element 8: Supply in BCF in 1995
Data Element 9: Well head gas price ($/MCF) in 1996
Data Element 10:        Supply in BCF in 1996
```

Table C-39

OUTPUT File: WELLSUMM.OUT (Location: \GSAM\EXPLPROD)
This file contains summary for the number of wells drilled by region, well type and resource type. It has been abridged to fit the width of the page.

```
Exploration Wells Drilled
                          1997      1998      1999      2000      2001      2002      2003
Pacific Offshore            .0        .0        .0        .0        .0        .0        .0
Pacific Onshore           31.2      22.3      28.4      31.0      17.1      13.1      38.0
San Juan                  22.6      17.4      28.2      31.6      38.3      35.9      42.7
Rockies Foreland         108.4     119.6     138.1     146.1     160.8     178.7     199.7
Williston                 32.3      31.5      36.7      42.7      47.2      38.5      42.5
Permian                  137.4     160.3     177.6     187.1     165.8      94.1      72.4
Mid-Continent            209.0     146.4     191.5     170.0     100.4      42.4      33.2
Arkla-East Texas          68.5      60.3      60.5      61.4      74.1      50.4      48.1
Texas Gulf Coast         205.8     176.2     216.9     221.7     220.3     198.5     186.1
Gulf of Mexico-West       60.3      67.5      73.8      80.9      90.6     101.0     111.5
Gulf of Mexico-Cntr       90.8      87.1      86.6     125.3     108.8     126.0     129.2
Norphlet                    .0        .0        .0        .0        .0        .0        .0
Gulf of Mexico-East         .0        .0        .0        .0        .0        .0        .0
So-Louisiana              66.0      71.9      80.4      86.3      92.9     105.0     116.6
MAFLA Onshore             70.9      50.9      50.8      54.1      43.2      17.9      16.5
Mid-West                  58.4      32.2      53.0      61.7      85.3      81.3      64.2
Appalachia               161.4      79.7      99.1      85.7      80.9      50.9      75.1
Alberta                  992.2    1588.9    2012.1    2096.5    1382.5     326.3     344.1
British Columbia          64.6     133.3     261.0     272.8     346.8     351.3     394.4
North Alaska                .0        .0        .0        .0        .0        .0        .0
MacKenzie Delta             .0        .0        .0        .0        .0        .0        .0
Atlantic Offshore           .0        .0        .0        .0        .0        .0        .0
Mexico-Supply               .0        .0        .0        .0        .0        .0        .0

Total U.S.              1322.8    1123.2    1321.5    1385.6    1325.7    1133.6    1175.9
Conventional             659.0     577.7     686.4     699.0     612.5     449.3     435.7
Tight                    234.2     196.9     209.3     211.3     233.5     241.4     301.2
Associate-Dissolved         .0        .0        .0        .0        .0        .0        .0
Natural Fracture          10.1      11.0      12.3      17.6      21.7      11.3       9.0
Water Drive               42.6      47.5      52.4      53.4      48.1      36.5      35.0
Coal and Shale           225.8     135.4     200.7     198.1     210.5     168.2     154.3
Analyzed Resource        151.1     154.6     160.4     206.2     199.4     227.0     240.8
```

```
Total Canada           1056.8    1722.2    2273.1    2369.3    1729.3     677.6     738.5
Conventional            827.0    1399.1    2107.9    1968.0    1582.3     579.8     603.7
Tight                     2.1       1.9       1.7       4.5       4.5       2.5       2.3
Associate-Dissolved        .0        .0        .0        .0        .0        .0        .0
Natural Fracture        227.7     321.1     144.7     353.6      97.2      45.8      82.1
Water Drive                .0        .0        .0        .0        .0        .0        .0
Coal and Shale             .0        .0      18.7      43.2      45.3      49.5      50.4
Analyzed Resource          .0        .0        .0        .0        .0        .0        .0
```

Table C-40

OUTPUT File: EXPLWLS.OUT (Location: \GSAM\EXPLPROD)
This file contains summary for the number of exploration wells drilled by play for the period 1997-2020. It has been abridged to fit the width of the page.

```
#   Play                    1997    1998    1999    2000    2001    2002    2003
     Exp. Wells Drilled ---->
.
.
.
461  4717P                 4.177   3.840   3.563   3.316   3.096   2.898   2.721
462  4718F                  .155    .000    .154    .663    .601    .546    .498
463  4718P                12.983  16.638  15.077  15.866  14.581  13.478  12.516
464  4719F                  .000    .000    .000    .000    .000    .000    .000
465  4719P                 1.890   1.728   1.586   1.453   1.330   1.216   1.111
466  4720F                  .000    .000    .000    .000    .197    .196    .194
467  4720P                10.608   9.749  20.367  19.276  18.268  14.950  14.311
468  4721F                  .000    .000    .000    .000    .000    .000    .000
469  4721P                  .000    .000    .000    .000    .000    .000    .000
470  4722F                  .000    .000    .000    .000    .000    .000    .000
471  4722P                16.455  11.682  15.730  15.397  28.925  27.581  26.316
472  4723F                  .000    .000    .000    .000    .000    .000    .000
473  4723P                21.107  23.527  15.365  22.740  22.085  21.460  20.865
474  4724F                  .000    .000    .000    .000    .000    .000    .000
475  4724P                22.234  13.945  18.232  16.362  14.725  13.287  12.016
476  4725F                  .000    .000    .000    .000    .000    .000    .000
477  4725P                 6.297   2.483   5.444   4.966   4.546   4.059   3.790
478  4726F                  .515    .443    .383    .331    .285    .246    .212
479  4726P                20.799  18.321  16.120  15.556  13.230  11.172   9.372
480  4727F                  .000    .000    .000    .000    .000    .000    .848
481  4727P                23.327  27.796  45.315  42.932  45.017  47.231  53.412
482  4728F                  .000    .000    .000    .000    .000    .000    .000
483  4728P                  .000    .000    .000    .000    .000    .000    .000
484  4729F                  .000    .000    .000    .000    .000    .000    .000
485  4729P                24.118  22.576  21.283  22.688  21.259  19.941  16.455
486  4730F                  .000    .000    .000    .000    .000    .000    .000
487  4730P                 2.634   2.365   2.133   1.920   2.072   1.512   1.356
488  4731F                  .000    .000    .000    .000    .000    .000    .000
489  4731P                 5.668   2.660   5.154   4.898   4.667   2.139   2.066
```

```
490  4732F                      .515    .098    .427    .369    .318    .274    .236
491  4732P                    13.019   7.173  11.593  10.824  10.112   9.451   8.837
492  4733F                      .000    .000    .000    .000    .000    .000    .000
493  4733P                      .000    .000    .000    .000    .000    .000    .000
494  4734F                      .000    .000    .000    .000    .000    .000    .000
495  4734P                     4.052   3.678   3.367   3.088   2.837   2.610   3.080
496  4735F                      .000    .000    .000    .120    .118    .115    .113
497  4735P                     6.711   4.643   6.019   5.705   5.416   5.150   4.903
498  4736F                      .000    .000    .000    .000    .123    .123    .123
.
.
.
```
(continues down with other region

APPENDIX D
DEMAND AND INTEGRATING MODULE FILES

CONTENTS

D-35	GASPRC.HIS
D-36	IND_EI.SPC
D-37	IND_ELS.SPC
D-38	IND_LD.SPC
D-39	IND_PRC.SPC
D-40	LNG.SPC
D-41	NOX.SPC
D-42	NUGS.SPC
D-43	ODUALS.SPC
D-44	ODUALS.STR
D-45	POP_GRP.SPC
D-46	PROC.SPC
D-47	PROPANE.SPC
D-48	RES_EFF.SPC
D-49	RES_ELS.SPC
D-50	RES_LD.SPC
D-51	RES_POP.SPC
D-52	RES_PRC.SPC
D-53	SOX.SPC
D-54	STORVALS.SPC
D-55	SUP_LD.SPC
D-56	SUPPLY.SPC
D-57	WEATHER.SPC

OUTPUT FILES

D-58	GASPRC.NEW
D-59	GSAMSLN.FLE
D-60	GSAMSLN.RPT
D-61	GSAMSLN.STA
D-62	GSAMSLN.STC
D-63	GSAMSLN.SEA
D-64	GSAMSLN.SUP
D-65	GASALL.PRT

Table D-1

Input Data File: CTGPRC.DAT (Location: \GSAM\DEMDINTG)
This file is an input and an output of the Demand and Integrating Modules. It shows the city gate (wholesale) natural gas price in $/MCF by GSAM demand region by year of analysis. The years in the file are 1998-2020. The file has been abridged to fit the width of the page.

```
New England            3.43   3.18   2.78   2.78   2.68   2.75   2.68   2.59   2.67   2.70   2.77
Middle Atlantic        3.41   3.12   2.67   2.65   2.55   2.61   2.56   2.47   2.54   2.55   2.61
South Atlantic         3.38   3.11   2.46   2.38   2.39   2.49   2.47   2.39   2.39   2.36   2.45
Florida                2.59   2.59   2.52   2.50   2.51   2.61   2.63   2.68   2.64   2.56   2.77
East South Central     3.39   3.09   2.39   2.30   2.32   2.42   2.32   2.25   2.12   2.03   2.09
East North Central     3.53   3.22   2.48   2.40   2.41   2.52   2.44   2.37   2.27   2.24   2.31
West South Central     2.81   2.60   2.24   2.19   2.19   2.27   2.18   2.10   1.97   1.90   1.96
West North Central     3.43   3.08   2.32   2.21   2.18   2.28   2.20   2.13   2.00   1.93   1.98
Mountain South         3.05   2.82   2.30   2.25   2.25   2.34   2.24   2.21   2.02   1.91   1.98
Mountain North         2.72   2.57   2.19   2.00   1.98   1.99   1.92   1.89   1.72   1.62   1.68
California             2.86   2.71   2.39   2.37   2.39   2.47   2.37   2.33   2.13   2.10   2.24
Pacific Northwest      2.91   2.81   2.37   2.26   2.24   2.25   2.16   2.12   1.94   1.84   1.89
Canada-East            3.23   3.07   2.66   2.63   2.53   2.62   2.56   2.48   2.52   2.55   2.60
Western Canada         2.58   2.51   2.28   2.20   2.19   2.19   2.12   2.10   1.96   1.92   1.98
BC-Demand              2.78   2.72   2.34   2.24   2.23   2.24   2.15   2.13   1.95   1.91   1.97
Mexico-Demand          2.90   2.69   2.23   2.19   2.17   2.27   2.18   2.10   1.94   1.86   1.92
```

Table D-2

Input Data File: SROM.IN (Location: \GSAM\DEMDINTG)
This file inputs the properties of storage reservoirs. It has been shortened to fit to one page in the Appendix.

```
Dictionary
LC:   Levelized Investment Cost, $/MCF
FOM:  Fixed O&M Cost, $/MCF
VOM:  Variable O&M Cost, $/MCF
MERi: Maximum Extraction Rate for Season i, % of Working Gas Per Day
MIR:  Maximum Injection Rate for Season 4, % of Working Gas Per Day

storage id  storage   Total   Tot.   Yr.  Fuel   =========== OPTION 1 PARAMETERS ============    ======== OPTION 2 PARAMETERS =========     ===== OPTION 3 PARAMETERS =====   Storage Region
             first     W.G.   Norm.        Used    LC     FOM    VOM    MER1  MER2   MER3   MIR     LC     FOM    VOM   MER1   MER2   MIR      LC    FOM    VOM   MER1   MIR        Name
              Yr.     (MMcf)  B.G.      inj/ext
                                          (%)
13000000001  1990  165997.0   .000  20.  1.00   0.70   0.05   0.02   0.82   0.82   0.82   0.41   0.70   0.05   0.02   0.82   0.82   0.41   0.70   0.05   0.02   0.82   0.41 Canada-East
14000000001  1990   91248.0   .000  20.  1.00   0.70   0.04   0.02   0.82   0.82   0.82   0.41   0.70   0.04   0.02   0.82   0.82   0.41   0.70   0.04   0.02   0.82   0.41 Western Canada
02706719218  1947     491.5   .000  20.  2.22    .93    .07    .03   1.04    .96    .77    .28    .84    .07    .03    .98    .77    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706720448  1991   11556.9   .000  20.  2.22    .50    .06    .08   1.45   1.20    .68    .28    .46    .06    .07   1.24    .68    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721201  1947    3720.1   .000  20.  2.22    .51    .02    .00   1.17   1.05    .74    .28    .46    .02    .00   1.07    .74    .28    .42    .02    .00    .83    .28 Middle Atlantic
02706721203  1947     299.9   .000  20.  2.22    .51    .02    .00    .91    .87    .81    .28    .46    .02    .00    .87    .81    .28    .42    .02    .00    .83    .28 Middle Atlantic
02706721204  1971     510.5   .000  20.  2.22    .51    .02    .00    .96    .91    .79    .28    .46    .02    .00    .92    .79    .28    .42    .02    .00    .83    .28 Middle Atlantic
02706721205  1957   18343.3   .000  20.  2.22    .50    .06    .08   1.44   1.20    .68    .28    .46    .06    .07   1.24    .68    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721217  1950    2618.5   .000  20.  2.22    .93    .07    .03   1.25   1.10    .72    .28    .84    .07    .03   1.12    .72    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706721219  1980    9468.6   .000  20.  2.22    .93    .07    .03   1.49   1.23    .67    .28    .84    .07    .03   1.27    .67    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706721220  1972    1582.1   .000  20.  2.22    .93    .07    .03   1.06    .98    .77    .28    .84    .07    .03    .99    .77    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706721221  1980    4179.7   .000  20.  2.22    .93    .07    .03   1.18   1.05    .74    .28    .84    .07    .03   1.07    .74    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706721257  1969     953.0   .000  20.  2.22    .50    .06    .08   1.20   1.07    .74    .28    .46    .06    .07   1.09    .74    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721258  1972    5942.7   .000  20.  2.22    .51    .02    .00   1.29   1.12    .72    .28    .46    .02    .00   1.15    .72    .28    .42    .02    .00    .83    .28 Middle Atlantic
02706721259  1972     925.1   .000  20.  2.22    .51    .02    .00   1.29   1.12    .72    .28    .46    .02    .00   1.15    .72    .28    .42    .02    .00    .83    .28 Middle Atlantic
02706721264  1963   51560.2   .000  20.  2.22    .50    .06    .08   1.48   1.22    .68    .28    .46    .06    .07   1.26    .68    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721265  1961   28859.4   .000  20.  2.22    .50    .06    .08   1.54   1.23    .67    .28    .46    .06    .07   1.28    .67    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721266  1953   20717.0   .000  20.  2.22    .50    .06    .08   1.70   1.32    .63    .28    .46    .06    .07   1.38    .63    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721267  1959   54437.6   .000  20.  2.22    .50    .06    .08   1.46   1.19    .69    .28    .46    .06    .07   1.24    .69    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721270  1951   23731.5   .000  20.  2.22    .50    .06    .08   1.92   1.42    .60    .28    .46    .06    .07   1.50    .60    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721271  1951   17840.1   .000  20.  2.22    .50    .06    .08   1.42   1.19    .69    .28    .46    .06    .07   1.23    .69    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721272  1948    2297.4   .000  20.  2.22    .50    .06    .08   1.25   1.10    .73    .28    .46    .06    .07   1.12    .73    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721287  1953   11909.9   .000  20.  2.22    .93    .07    .03   1.23   1.08    .73    .28    .84    .07    .03   1.11    .73    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706721296  1959     833.4   .000  20.  2.22    .93    .07    .03   1.01    .95    .78    .28    .84    .07    .03    .96    .78    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706721304  1971    9058.0   .000  20.  2.22    .50    .06    .08   1.34   1.14    .71    .28    .46    .06    .07   1.17    .71    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706725268  1951   60961.6   .000  20.  2.22    .50    .06    .08   1.11   1.01    .76    .28    .46    .06    .07   1.03    .76    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706725275  1947    1662.1   .000  20.  2.22    .64    .02    .02   1.17   1.05    .74    .28    .58    .01    .02   1.07    .74    .28    .53    .01    .02    .83    .28 Middle Atlantic
02706725300  1947     505.0   .000  20.  2.22    .50    .06    .08   1.02    .96    .78    .28    .46    .06    .07    .97    .78    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706725306  1947     557.7   .000  20.  2.22    .50    .06    .08   1.53   1.22    .67    .28    .46    .06    .07   1.27    .67    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706725356  1947     872.1   .000  20.  2.22    .51    .02    .00   1.40   1.18    .69    .28    .46    .02    .00   1.22    .69    .28    .42    .02    .00    .83    .28 Middle Atlantic
02706732208  1951     843.1   .000  20.  2.22    .93    .07    .03    .98    .92    .79    .28    .84    .07    .03    .93    .79    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706732209  1952    7381.0   .000  20.  2.22    .93    .07    .03   1.27   1.11    .72    .28    .84    .07    .03   1.14    .72    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706732210  1948    1058.6   .000  20.  2.22    .93    .07    .03    .97    .92    .79    .28    .84    .07    .03    .93    .79    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706732211  1950     130.4   .000  20.  2.22    .93    .07    .03   1.10   1.01    .76    .28    .84    .07    .03   1.02    .76    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706732212  1949     462.7   .000  20.  2.22    .93    .07    .03    .98    .93    .79    .28    .84    .07    .03    .94    .79    .28    .77    .06    .03    .83    .28 Middle Atlantic
```

.

.

.
(continues with other storage reservoirs)

Intended Uses of SROM.IN
The values in SROM.IN are calculated from the Storage Reservoir Performance Module (SRPM). However, this file itself may be altered to reflect different beliefs about specific storage reservoirs or to conduct a sensitivity analysis.

Table D-3

Input Data File: BARRIER.CFG (Location: \GSAM\DEMDINTG)
This file is used to specify the run settings of the Integrating Module's LP solver.

```
m_r = 95000
m_c = 585000
m_m = 3064000
densecol = 0
border=2
input
minim -b
fprint
basisout
quit
```

Table D-4

Input Data File: COAL_PR.SPC (Location: \GSAM\DEMDINTG)
This file contains coal price specifications by year in analysis.

```
c** Coal Prices File          annual  annual  annual  annual  annu
c**               base        growth  growth  growth  growth  grow
c**               price       rate %  rate %  rate %  rate %  rate
c**               1995        [1996,  (2000,  (2005,  (2010,  (2015,
c**               ($/MCF)      2000]   2005]   2010]   2015]   2020]
"New England       "  1.57     1.960  -1.305  -0.499  -1.034  -0.951
"Middle Atlantic   "  1.25    -0.813  -0.505  -0.172  -0.877  -1.105
"South Atlantic    "  1.38    -0.292  -1.051  -0.628  -1.146  -1.394
"Florida           "  1.59    -2.793  -1.648  -0.963  -0.840  -1.057
"East South Central"  1.09     2.941  -1.137  -0.169  -1.216  -1.294
"East North Central"  1.07     0.737  -1.105  -0.384  -1.193  -1.269
"West South Central"  0.75    10.933  -1.303  -0.687  -1.445  -1.557
"West North Central"  0.58    10.833  -1.269  -0.221  -1.137  -1.453
"Mountain South    "  1.02     2.249  -2.200  -0.595  -0.821  -0.857
"Mountain North    "  0.78     0.255  -1.839  -0.562  -1.170  -1.243
"California        "  9.99     0.000   0.000   0.000   0.000   0.000
"Pacific Northwest "  0.04    88.024  -1.993  -0.959  -1.527  -1.370
"Canada-East       "  2.09    -1.785  -1.400  -2.110  -1.950  -2.010
"Western Canada    "  0.38    -0.532  -1.105  -0.578  -1.205  -0.633
"BC-Demand         "  9.99     0.000   0.000   0.000   0.000   0.000
"Mexico-Demand     "  1.60    -1.414  -1.954  -1.685  -0.820  -0.681
```

Explanation

COAL_PR.SPC specifies the coal prices in $/MCF equivalent by year. Coal competes with gas and other fuels to determine product-specific demand in the electrical generation sector.

Intended Uses

Changing the price of coal in a time period can affect the calculation of electric utilities' demand for gas.

Table D-5

File: COM_DEM.SPC (Location: \GSAM\DEMDINTG)
This file contains commercial demand specifications by GSAM demand region.

```
c** commercial  demand file  annual annual  annual  annual  annual
c**                base    growth growth  growth  growth  growth  inter
c**                demd    rate % rate %  rate %  rate %  rate %  demand
c**                1995    [1996, (2000,  (2005,  (2010,  (2015,  share
c**                (BCF)   2000]  2005]   2010]   2015]   2020]   %
"New England       " 143.8    0.727  1.064   1.034   0.902   0.908  10
"Middle Atlantic   " 514.1    0.332  0.500   0.528   0.361   0.358  10
"South Atlantic    " 265.2    2.194  1.440   1.468   1.256   1.261  10
"Florida           "  40.4    2.180  1.467   1.445   1.275   1.233  10
"East South Central" 136.0    0.292  0.601   0.666   0.445   0.435  10
"East North Central" 740.6   -0.351  0.444   0.553   0.407   0.409  10
"West South Central" 300.5    2.440  1.192   1.194   1.102   1.101  10
"West North Central" 299.7    0.659  0.994   0.947   0.876   0.872  10
"Mountain South    "  52.1    2.133  1.279   1.233   1.016   0.995  10
"Mountain North    " 143.2    2.141  1.274   1.231   1.011   1.013  10
"California        " 278.8    1.846  1.020   1.042   0.825   0.825  10
"Pacific Northwest "  65.0    1.839  1.018   1.046   0.823   0.837  10
"Canada-East       " 279.2    0.014  0.907   0.588   0.584   0.581  10
"Western Canada    "  98.9    1.069  1.143   1.151   1.137   1.138  10
"BC-Demand         "  58.5    3.176  2.530   2.056   2.037   2.063  10
"Mexico-Demand     "   1.4    2.707  7.528   6.828   7.992   7.348  10
```

Explanation of COM DEM.SPC

The share of **demand that is interruptible (**inter demand share**)** indicates the share of the commercial sector which has interruptible contracts with their gas supplier.

The base quantity for the commercial demand is the commercial demand (in Bcf) for the given time period. The calculation for total commercial demand for a region and time period uses the base demand and adjusts it according to the base price, the current gas price, the population (if its elasticity was not 0), and the GRP.

Intended Uses of COM DEM.SPC

The commercial demand file can be altered to reflect assumptions about the commercial demand specifications.

Table D-6

Input Data File: DISTIND.SPC (one of several of this type)
(Location: \GSAM\DEMDINTG)
This is one of six files which shows the regional markups by fuel type by sector.

```
c*** regional margins for distillate and the industrial sector in $/MCF
c***                 1994  1995  2000  2005  2010  2015  2020
New England          0.86  0.86  0.86  0.86  0.86  0.86  0.86
Middle Atlantic      0.62  0.62  0.62  0.62  0.62  0.62  0.62
South Atlantic       0.46  0.46  0.46  0.46  0.46  0.46  0.46
Florida              0.62  0.62  0.62  0.62  0.62  0.62  0.62
East South Central   0.09  0.09  0.09  0.09  0.09  0.09  0.09
East North Central   0.70  0.70  0.70  0.70  0.70  0.70  0.70
West South Central   0.53  0.53  0.53  0.53  0.53  0.53  0.53
West North Central   0.39  0.39  0.39  0.39  0.39  0.39  0.39
Mountain South       0.01  0.01  0.01  0.01  0.01  0.01  0.01
Mountain North       0.25  0.25  0.25  0.25  0.25  0.25  0.25
California           0.36  0.36  0.36  0.36  0.36  0.36  0.36
Pacific Northwest    1.08  1.08  1.08  1.08  1.08  1.08  1.08
Canada-East          0.00  0.00  0.00  0.00  0.00  0.00  0.00
Western Canada       0.00  0.00  0.00  0.00  0.00  0.00  0.00
BC-Demand            0.00  0.00  0.00  0.00  0.00  0.00  0.00
Mexico-Demand        9.99  9.99  9.99  9.99  9.99  9.99  9.99
```

Explanation

For the calculation of industrial and electric utility demand, the regional margins account for cost (and the effect on price) of transportation of a petroleum product from a Gulf Coast refinery to the region. Three different petroleum products (distillate, low sulfur residual, and high sulfur residual) are subject to demand by two sectors (industrial and electric generation). There are six files of this type: DISTIND.SPC, DISTELE.SPC, 1PCTIND.SPC, 1PCTELE.SPC, 3PCTIND.SPC, and 3PCTELE.SPC.

Intended Uses

The user may change any of the markups as desired. It will ultimately affect the regional price of whichever product is changed, for either the electric utility sector or the industrial sector.

Table D-7

Input Data File: DMN_SEC.SPC (Location: \GSAM\DEMDINTG)
This file contains demand sector markup specifications. It has been shortened to fit to one page in the Appendix.

```
c** City-gate to burner tip mark-ups by sector, region, and year
c**                                 base    growth growth  growth  growth  growth
c**                                 value   rate % rate %  rate %  rate %  rate %
c**                                 1995    [1996, (2000,  (2005,  (2010,  (2015,
c**                                 ($/Mcf)  2000]  2005]   2010]   2015]   2020]
"New England      "  "Residential"   5.88   -5.00  -5.00   -5.00   -5.00   -5.00
"Middle Atlantic  "  "Residential"   4.76   -5.00  -5.00   -5.00   -5.00   -5.00
"South Atlantic   "  "Residential"   4.28   -5.00  -5.00   -5.00   -5.00   -5.00
"Florida          "  "Residential"   6.74   -5.00  -5.00   -5.00   -5.00   -5.00
"East South Central"  "Residential"  3.28   -5.00  -5.00   -5.00   -5.00   -5.00
"East North Central"  "Residential"  2.89   -5.00  -5.00   -5.00   -5.00   -5.00
"West South Central"  "Residential"  3.22   -5.00  -5.00   -5.00   -5.00   -5.00
"West North Central"  "Residential"  2.47   -5.00  -5.00   -5.00   -5.00   -5.00
"Mountain South   "  "Residential"   4.07   -5.00  -5.00   -5.00   -5.00   -5.00
"Mountain North   "  "Residential"   2.64   -5.00  -5.00   -5.00   -5.00   -5.00
"California       "  "Residential"   4.03   -5.00  -5.00   -5.00   -5.00   -5.00
"Pacific Northwest "  "Residential"  2.95   -5.00  -5.00   -5.00   -5.00   -5.00
"Canada-East      "  "Residential"   5.88   -5.00  -5.00   -5.00   -5.00   -5.00
"Western Canada   "  "Residential"   2.47   -5.00  -5.00   -5.00   -5.00   -5.00
"BC-Demand        "  "Residential"   2.47   -5.00  -5.00   -5.00   -5.00   -5.00
"Mexico-Demand    "  "Residential"   3.22   -5.00  -5.00   -5.00   -5.00   -5.00
"New England      "  "Commercial "   4.09   -5.00  -5.00   -5.00   -5.00   -5.00
"Middle Atlantic  "  "Commercial "   3.29   -5.00  -5.00   -5.00   -5.00   -5.00
"South Atlantic   "  "Commercial "   2.94   -5.00  -5.00   -5.00   -5.00   -5.00
"Florida          "  "Commercial "   2.68   -5.00  -5.00   -5.00   -5.00   -5.00
"East South Central"  "Commercial "  2.51   -5.00  -5.00   -5.00   -5.00   -5.00
"East North Central"  "Commercial "  2.37   -5.00  -5.00   -5.00   -5.00   -5.00
"West South Central"  "Commercial "  1.87   -5.00  -5.00   -5.00   -5.00   -5.00
"West North Central"  "Commercial "  1.65   -5.00  -5.00   -5.00   -5.00   -5.00
"Mountain South   "  "Commercial "   2.48   -5.00  -5.00   -5.00   -5.00   -5.00
"Mountain North   "  "Commercial "   1.94   -5.00  -5.00   -5.00   -5.00   -5.00
"California       "  "Commercial "   3.26   -5.00  -5.00   -5.00   -5.00   -5.00
"Pacific Northwest "  "Commercial "  2.04   -5.00  -5.00   -5.00   -5.00   -5.00
"Canada-East      "  "Commercial "   4.09   -5.00  -5.00   -5.00   -5.00   -5.00
"Western Canada   "  "Commercial "   1.65   -5.00  -5.00   -5.00   -5.00   -5.00
"BC-Demand        "  "Commercial "   1.65   -5.00  -5.00   -5.00   -5.00   -5.00
"Mexico-Demand    "  "Commercial "   1.87   -5.00  -5.00   -5.00   -5.00   -5.00
```

.

.

.
(continues with other sectors)

<u>Explanation</u>

DMN_SEC.SPC contains the markups that distributors will place on gas in the various regions and sectors
.

Table D-8

File: EU_DEM.SPC (Location: \GSAM\DEMDINTG)
This file contains electric utility demand specifications. Its width has been abridged to fit to one page in the Appendix.

```
New England         EX-CAP  NUCLEAR   6.1  22.5  50.1  21.3   6.53   6.24   5.51   5.48   5.44   2.81   0.20
Middle Atlantic     EX-CAP  NUCLEAR   6.3  22.5  49.8  21.4  17.20  17.17  17.12  17.07  17.02   9.61   2.21
Florida             EX-CAP  COAL      6.2  23.9  49.4  20.5   9.38   9.08   8.35   8.35   8.35   7.96   7.57
East North Central  EX-CAP  COAL      5.9  21.8  50.7  21.6  76.58  78.68  83.94  83.64  83.33  72.10  60.88
Pacific Northwest   EX-CAP  HYDRO/OT  6.4  22.2  49.1  22.3  29.25  29.17  28.96  28.96  28.96  28.96  28.96
California          EX-CAP  HYDRO/OT  6.4  22.5  49.9  21.2  13.68  12.99  11.26  11.26  11.26  11.26  11.26
Canada-West         EX-CAP  COMB CYL  6.4  22.7  49.8  21.1   0.30   0.30   0.30   0.30   0.30   0.30   0.30
Mexico-Demand       EX-CAP  COMB CYL  6.2  23.9  49.4  20.5   1.00   1.00   1.00   1.00   1.00   1.00   1.00
New England         EX-CAP  O/G LS-R  6.1  22.5  50.1  21.3   5.12   4.89   4.32   4.27   4.22   3.53   2.84
Middle Atlantic     EX-CAP  O/G LS-R  6.3  22.5  49.8  21.4  12.44  11.35   8.64   8.55   8.45   7.03   5.61
South Atlantic      EX-CAP  O/G LS-R  6.4  23.1  49.3  21.2   4.28   3.78   2.54   2.51   2.48   2.07   1.66
Florida             EX-CAP  O/G LS-R  6.2  23.9  49.4  20.5  10.55  10.75  11.26  11.26  11.26  10.39   9.53

Mountain 2          EX-CAP  O/G HS-R  6.4  22.2  49.1  22.3   0.16   0.13   0.07   0.07   0.07   0.06   0.05
Canada-West         EX-CAP  O/G HS-R  6.4  22.7  49.8  21.1   0.00   0.00   0.00   0.00   0.00   0.00   0.00

Mexico-Demand       EX-CAP  O/G HS-R  6.2  23.9  49.4  20.5   0.00   0.00   0.00   0.00   0.00   0.00   0.00
New England         EX-CAP  O/G DIST  6.1  22.5  50.1  21.3   2.54   2.54   2.54   2.54   2.54   2.54   2.54
Middle Atlantic     EX-CAP  O/G DIST  6.3  22.5  49.8  21.4   8.44   8.44   8.44   8.44   8.44   8.44   8.44
Middle Atlantic     TOT-ELEC         6.3  22.5  49.8  21.4 314.25 324.29 350.82 369.68 389.57 427.10 468.26 0.13 1.61 0.00 0.00 0.00 0.00 0.00 1.08 0.94 0.00 0.00 0.00 0.00 0.00 1.14
South Atlantic      TOT-ELEC         6.4  23.1  49.3  21.2 421.12 437.27 480.41 513.29 548.43 613.12 685.45 0.97 1.02 0.00 0.00 0.00 0.00 0.00 0.76 1.17 0.00 0.00 0.00 0.00 0.00 1.18
Florida            TOT-ELEC         6.2  23.9  49.4  20.5 134.58 136.60 141.77 149.78 158.24 173.72 190.72 0.00 1.71 0.00 0.00 0.00 0.00 0.00 0.00 1.71 0.00 0.00 0.00 0.00 0.00 0.95
East North Central  TOT-ELEC         5.9  21.8  50.7  21.6 492.75 503.13 530.06 558.79 589.08 647.23 711.13 0.39 1.43 0.00 0.00 0.00 0.00 0.00 1.28 0.80 0.00 0.00 0.00 0.00 0.00 0.91
East South Central  TOT-ELEC         6.4  23.1  49.3  21.2 268.86 278.79 305.24 326.04 348.25 389.17 434.91 0.97 1.02 0.00 0.00 0.00 0.00 0.00 0.76 1.17 0.00 0.00 0.00 0.00 0.00 1.18
West North Central  TOT-ELEC         6.4  22.7  49.8  21.1 215.86 222.79 241.10 254.08 267.76 294.21 323.26 0.80 1.14 0.00 0.00 0.00 0.00 0.00 1.65 0.54 0.00 0.00 0.00 0.00 0.00 0.97
West South Central  TOT-ELEC         6.7  24.7  48.3  20.3 385.10 398.77 435.11 460.26 486.87 535.91 589.89 0.00 1.71 0.00 0.00 0.00 0.00 0.00 0.10 1.63 0.00 0.00 0.00 0.00 0.00 1.10
```

Description of File: EU_DEM.SPC

Data Element	Description	Format
1	Demand region	A20
2	Demand flag	A8
3	Fuel name	A8
4-7	Variables used in calculating scaling factor	F5.2, 1X
8-14	Generating capacity for time periods specified	F6.2, 1X
15-42	(TOT-ELEC demand entries only)	F5.2
43-49	(All other entries)	F5.2
	Maximum plant utilization rate	

Explanation of EU_DEM.SPC

The four entries for the **scaling factor** correspond to four seasons modeled in GSAM's electricity generation calculations. These mean that, for instance in New England, 6.1% of electricity generation will occur in the first modeled season, 22.5% in the second, and so forth. These numbers should sum to one.

The **generating capacity** is the most electricity that can be generated in a given region for a given time, if the utilization rate is 100%.

For the TOT-ELEC demand entries only, the **load profile** is for time *ab*, where a is the gas season (1-2) and b is the electric utility season (1-4). An average day will have a load factor of 1. For example, in Middle Atlantic, days that are in gas season 1 and EU season 1 will have a factor of 0.13

For the non- TOT-ELEC entries, the last set of numbers gives the **maximum plant utilization rate**. For New England in 1993, the electric utility plants will be utilized at a rate of 67% in the above file.

Intended Uses of EU_DEM.SPC

The parameters in the EU_DEM.SPC can be changed, but note that the format must be the same. The capacity and utilization rate are the parameters that are most often changed. To expand electricity generation from a given fuel, either of these numbers can be increased, and to reduce electric generation from a fuel type, either may be decreased.

Table D-9

File: EU_GEN.SPC (Location: \GSAM\DEMDINTG)
This file contains electric utility efficiency and cost data.

```
Existing                     New
======== ========================== ========================== =====  =====================
Syr Eyr    Cap. Fixed  Var.    Heat   Cap. Fixed  Var.    Heat  Cap.    New Capacity Util.
           Cost   O&M   O&M    Rate   Cost   O&M   O&M    Rate  Rate  Seas  Seas  Seas  Seas
Fuel       $/MW  $/MW  $/MW Btu/Kwh   $/MW  $/MW  $/MW Btu/Kwh     %     1     1     1     1
======== ====== ===== ===== ======= ====== ===== ===== ======= =====  =====================
1993 2000
NUCLEAR  1895.0 91.62  0.50 10521.0 9988.2 54.07  0.40 10400.0 12.10  0.71  0.71  0.71  0.71
COAL     1188.0 26.00  2.80  9343.0 1322.0 33.00  2.50  9216.0 12.10  0.71  0.71  0.71  0.71
HYDRO/OT 9999.9  4.30  4.30 11000.0 9999.9 10.00  6.00 11000.0 12.10  0.59  0.59  0.59  0.59
O/G LS-R 1180.0  7.10  7.00  9680.0 1180.0  7.10  7.00  9680.0 12.10  0.85  0.85  0.85  0.85
O/G HS-R 1180.0  7.10  7.00  9680.0 1180.0  7.10  7.00  9680.0 12.10  0.85  0.85  0.85  0.85
COMB CYL  523.0 18.02  1.00  8000.0  608.0 17.10  1.10  6964.0 12.10  0.92  0.92  0.92  0.92
O/G DIST  364.0  3.04  1.00 12303.0  370.0  6.90  1.00 10953.0 12.10  0.92  0.92  0.92  0.92
======== ====== ===== ===== ======= ====== ===== ===== ======= =====  =====================
2001 2002
NUCLEAR  1895.0 91.62  0.50 10521.0 9988.2 54.07  0.40 10400.0 12.10  0.71  0.71  0.71  0.71
COAL     1188.0 26.00  2.80  9343.0 1322.0 33.00  2.50  9216.0 12.10  0.71  0.71  0.71  0.71
HYDRO/OT 9999.9  4.30  4.30 11000.0 9999.9 10.00  6.00 11000.0 12.10  0.59  0.59  0.59  0.59
O/G LS-R 1180.0  7.10  7.00  9680.0 1180.0  7.10  7.00  9680.0 12.10  0.85  0.85  0.85  0.85
O/G HS-R 1180.0  7.10  7.00  9680.0 1180.0  7.10  7.00  9680.0 12.10  0.85  0.85  0.85  0.85
COMB CYL  523.0 18.02  1.00  7938.7  585.0 17.10  1.10  6875.0 12.10  0.92  0.92  0.92  0.92
O/G DIST  364.0  3.04  1.00 12208.7  357.0  6.90  1.00 10834.0 12.10  0.92  0.92  0.92  0.92
======== ====== ===== ===== ======= ====== ===== ===== ======= =====  =====================
```

.

.

.
(continues with other years)

Explanation

The parameters in EU_GEN.SPC determine: when and how much new EU capacity is added (the cost variables) and the efficiency and productivity of EU plants (heat rate) by fuel type.

Intended Uses

EU_GEN.SPC is used when altering the electric generating sector's operating parameters. It changes the relative cost and efficiency of one fuel vs. another in competition for electricity generation. This will affect the overall demand for gas.

Table D-10

Input Data File: GEN_TML.SPC (Location: \GSAM\DEMDINTG)
This file contains specifications for running the module.

```
1998 2020 1998               ! GSAM begin year, end year, current year
  5  26  90 244   0   0   0  ! Number of Days per Gas Season
 18  73 183  91              ! Number of Days per EU Season
10.0                         ! Discount Rate
```

Table D-11

Input Data File: LINK_NDE.SPC (Location: \GSAM\DEMDINTG)
This file contains transportation link specifications. It has been shortened to fit to one page in the Appendix.

```
East South Central Appalachia         2361.0   56.2   0.00 1991 99999.9   84.3   0.00 1995  0.016  2.77 1.00
North Alaska       Alberta               0.0 1008.5   0.00 1991 99999.9 1008.5   0.00 1995  0.307  3.70 0.00
MacKenzie Delta    Alberta               0.0  745.7   0.00 1991 99999.9  745.7   0.00 1995  0.227  2.30 0.00
Rockies Foreland   California          730.0  245.0   0.00 1991  1230.0  245.0   0.00 2000  0.002  4.52 1.00
Pacific Onshore    California        50000.0   18.3   0.00 1991 99999.9   27.5   0.00 1995  0.000  2.00 1.00
Pacific Offshore   California          110.0   18.3   0.00 1991 99999.9   27.5   0.00 1995  0.000  2.00 1.00
Mountain 1         California         5078.0  107.5   0.00 1991 99999.9  134.4   0.00 1995  0.023  4.87 1.00
Pacific Northwest  California         1831.0   94.9   0.00 1991 99999.9   94.9   0.00 1995  0.060  1.16 1.00
Alberta            Canada-East        4053.0  278.0   0.00 1991 99999.9  236.0   0.00 2000  0.020  8.21 0.00
East North Central Canada-East        2030.0   90.0   0.00 1991 99999.9  135.0   0.00 1995  0.024  3.20 0.25
Alberta            Canada-West       50000.0   58.0   0.00 1991 99999.9   69.6   0.00 1995  0.005  1.11 0.00
Texas Gulf Coast   Arkla-East Texas   2527.0   66.2   0.00 1991 99999.9   99.3   0.00 1995  0.010  1.95 1.00
Mid-Continent      Arkla-East Texas    680.0  107.0   0.00 1991 99999.9  160.5   0.00 1995  0.005  3.80 1.00
Norphlet           MAFLA Onshore       600.0    9.8   0.00 1991 99999.9   14.6   0.00 1995  0.029  0.00 1.00
Gulf of Mexico-East So-Louisiana      7004.0   36.4   0.00 1991 99999.9  109.2   0.00 2005  0.004  0.74 1.00
Texas Gulf Coast   So-Louisiana       2653.0   55.1   0.00 1991 99999.9   82.8   0.00 1995  0.011  1.66 1.00
Lake Charles       So-Louisiana        700.0   36.0   0.00 1991 99999.9  576.0   0.00 1995  0.000  0.00 1.00
Appalachia         East North Central 3232.0   60.5   0.00 1991 99999.9   90.8   0.00 1995  0.015  1.94 1.00
Alberta            East North Central 2447.0  159.9   0.00 1991 99999.9  159.9   0.00 2030  0.016  4.31 0.25
Mid-West           East North Central 50000.0  57.4   0.00 1991 99999.9   86.2   0.00 1995  0.003  1.05 1.00
Arkla-East Texas   East North Central 2540.0   89.0   0.00 1991 99999.9  133.5   0.00 1995  0.029  3.10 1.00
East South Central East North Central 5973.0   81.6   0.00 1991 99999.9  122.5   0.00 1995  0.014  1.96 1.00
West North Central East North Central 4389.0   54.6   0.00 1991 99999.9   81.9   0.00 1995  0.054  3.38 1.00
MAFLA Onshore      East South Central 50000.0  65.8   0.00 1991 99999.9   98.9   0.00 1995  0.003  1.90 1.00
So-Louisiana       East South Central 50000.0  88.1   0.00 1991 99999.9  132.2   0.00 1995  0.023  2.41 1.00
So-Louisiana       Florida            1620.0  162.1   0.00 1991 99999.9  202.6   0.00 1995  0.036  4.43 1.00
South Atlantic     Florida              56.0   51.3   0.00 1991 99999.9  304.0   0.00 1995  0.014  0.06 1.00
West Florida       Florida           50000.0   18.4   0.00 1991 99999.9   27.6   0.00 1995  0.000  2.00 1.00
Williston          Rockies Foreland  50000.0   49.7   0.00 1991 99999.9   74.6   0.00 1995  0.016  1.60 1.00
Gulf of Mexico-West Texas Gulf Coast  2662.0   18.4   0.00 1991 99999.9   55.2   0.00 2000  0.000  0.61 1.00
Permian            Texas Gulf Coast   1600.0   36.5   0.00 1991 99999.9   54.8   0.00 1995  0.025  2.00 1.00
Mexico-Supply      Texas Gulf Coast   3375.0  100.0   0.00 1991 99999.9    2.4   0.00 2030  0.020  1.00 1.00
Permian            Mid-Continent      2483.0   50.6   0.00 1991 99999.9   75.9   0.00 1995  0.012  3.37 1.00
Appalachia         Middle Atlantic   50000.0   72.9   0.00 1991 99999.9  109.4   0.00 1995  0.023  2.38 1.00
Canada-East        Middle Atlantic    2072.0  117.3   0.00 1991 99999.9  146.6   0.00 2030  0.003  1.80 0.75
East North Central Middle Atlantic     990.0   73.7   0.00 1991 99999.9  110.7   0.00 1995  0.026  1.80 1.00
East South Central Middle Atlantic    3700.0  118.2   0.00 1991 99999.9  177.4   0.00 1995  0.113  2.71 1.00
South Atlantic     Middle Atlantic    2256.0   81.6   0.00 1991 99999.9  122.4   0.00 1995  0.012  2.29 1.00
Permian            Mountain 1         2313.0   53.0   0.00 1991 99999.9   66.2   0.00 1995  0.025  5.00 1.00
San Juan           Mountain 1        50000.0   48.1   0.00 1991 99999.9   96.2   0.00 1995  0.017  3.53 1.00
Rockies Foreland   Mountain 2        50000.0   77.3   0.00 1991 99999.9  116.0   0.00 1995  0.014  2.29 1.00
Mid-Continent      Mountain 2          385.0   94.8   0.00 1991 99999.9  142.2   0.00 1995  0.026  3.08 1.00
San Juan           Mountain 2          150.0   77.4   0.00 1991 99999.9  116.1   0.00 1995  0.001  1.13 1.00
Canada-East        New England          63.0   50.9   0.00 1991 99999.9   76.4   0.00 2030  0.011  1.54 0.75
Distrigas          New England         285.0  144.0   0.00 1991 99999.9  576.0   0.00 1995  0.000  0.00 1.00
Middle Atlantic    New England        2210.0   82.5   0.00 1991 99999.9  123.8   0.00 1995  0.011  4.99 1.00
Sable Island       New England           0.0  410.0   0.00 1991   400.0  410.0   0.00 2005  0.031  3.00 0.50
Alberta            Pacific Northwest  2426.0  108.5   0.00 1991 99999.9  135.6   0.00 2030  0.025  4.35 0.50
Rockies Foreland   Pacific Northwest   254.0  103.2   0.00 1991 99999.9  154.8   0.00 1995  0.001  1.51 1.00
Appalachia         South Atlantic    50000.0   75.0   0.00 1991 99999.9  112.5   0.00 1995  0.018  2.21 1.00
Cove Point         South Atlantic     1000.0  144.0   0.00 1991 99999.9  576.0   0.00 1995  0.000  0.00 1.00
So-Louisiana       South Atlantic     4909.0   96.7   0.00 1991 99999.9  145.1   0.00 1995  0.018  3.79 1.00
Elba Island        South Atlantic      540.0  144.0   0.00 1991 99999.9  576.0   0.00 1995  0.000  0.00 1.00
East South Central South Atlantic       97.0  132.5   0.00 1991 99999.9  198.8   0.00 1995  0.014  2.39 1.00
Rockies Foreland   San Juan            961.0  103.2   0.00 1991  1261.0  155.0   0.00 1995  0.001  1.51 1.00
Permian            San Juan            987.0   64.7   0.00 1991 99999.9   97.1   0.00 1995  0.016  3.12 1.00
Arkla-East Texas   West South Central 50000.0  68.3   0.00 1991 99999.9  102.6   0.00 1995  0.004  1.81 1.00
So-Louisiana       West South Central 16432.0  60.1   0.00 1991 99999.9   90.3   0.00 1995  0.004  2.00 1.00
Texas Gulf Coast   West South Central 50000.0  18.3   0.00 1991 99999.9   27.5   0.00 1995  0.050  2.00 1.00
Mid-Continent      West South Central 2909.0   71.0   0.00 1991 99999.9  106.5   0.00 1995  0.011  3.67 1.00
Permian            West South Central 50000.0  60.1   0.00 1991 99999.9   90.2   0.00 1995  0.004  2.00 1.00
Alberta            West North Central 1568.0   65.3   0.00 1991  2268.0   98.0   0.00 2000  0.006  2.50 0.40
Arkla-East Texas   West North Central  840.0   60.8   0.00 1991 99999.9   91.2   0.00 1995  0.006  1.79 1.00
Rockies Foreland   West North Central  547.0   91.2   0.00 1991  1147.0  136.8   0.00 2000  0.036  3.48 1.00
```

.

.

.

(Continues with other transportation links)

Description of File: LINK_NDE.SPC

Data Element	Description	Format
1	Supply region	A20
2	Demand region	A20
3	Current capacity of pipeline link (MMCF/day)	F7.1, 1X
4	Levelized investment costs (M$/MMCF/day capacity) for first transport capacity increment	F6.1, 1X
5	Annual fixed O&M costs (M$/MMCF/day capacity) for first transport capacity increment	F6.1, 1X
6	First year capacity becomes available	I4, 1X
7	New capacity of link (MMCF/day)	F7.1, 1X
8	Levelized investment costs (M$/MMF/day capacity) for second transport capacity increment	F6.1, 1X
9	Annual fixed O&M costs (M$/MMCF/day capacity) for second transport capacity increment	F6.1, 1X
10	First year capacity becomes available	I4, 1X
11	Annual variable costs ($/MCF) for transport capacity increments excluding fuel use	F6.3
12	Fuel use (%) for transport capacity increment	F5.2
13	Canada - USA transportation factor	F5.2

Explanation of LINK NDE.SPC

LINK_NDE.SPC contains all of the pipeline transportation specifications. GSAM will create new capacity along a link, if necessary demand conditions are met, economic requirements are fulfilled (w.r.t. the costs), and the year in which it is to be created is after the availability year.

The Canada-U.S.A. transportation factor is used only for reporting purposes. A zero means that all gas is used in Canada. A non-zero number (e.g., 0.25) means that 25% of the pipeline's gas is transported to the U.S., and the rest is used in Canada.

Intended Uses of LINK NDE.SPC

LINK_NDE.SPC is used to update or change the specifications on the transportation links. A pipeline expansion may be made more economic by lowering its costs, or may be made available earlier. The current and new capacity numbers may be changed, affecting the flows and the material balance among regions, with impacts on price, demand, and supply.

Table D-12

Input Data File: NODE.SPC (Location: \GSAM\DEMDINTG)
This file contains node specifications. This file appears in the exact version in the E&P Module.

```
New England          0  1   0.00  0.000
Middle Atlantic      0  2   0.00  0.000
South Atlantic       0  3   0.00  0.000
Florida              0  4   0.00  0.000
East South Central   0  5   0.00  0.000
East North Central   0  6   0.00  0.000
West South Central   0  7   0.00  0.000
West North Central   0  8   0.00  0.000
Mountain South       0  9   0.00  0.000
Mountain North       0 10   0.00  0.000
California           0 11   0.00  0.000
Pacific Northwest    0 12   0.00  0.000
Canada-East          0 13   0.00  0.000
Western Canada       0 14   0.00  0.000
BC-Demand            0 15   0.00  0.000
Mexico-Demand        0 16   0.00  0.000
Pacific Offshore     1  0   1.05  0.150
Pacific Onshore      2  0   1.05  0.150
San Juan             3  0   1.05  0.170
Rockies Foreland     4  0   1.05  0.120
Williston            5  0   1.05  0.150
Permian              6  0   1.05  0.100
Mid-Continent        7  0   1.05  0.100
Arkla-East Texas     8  0   1.05  0.070
Texas Gulf Coast     9  0   1.05  0.080
Gulf of Mexico-West 10  0   1.05  0.070
Gulf of Mexico-East 11  0   1.05  0.080
Norphlet            12  0   1.05  0.080
West Florida        13  0   1.05  0.080
So-Louisiana        14  0   1.05  0.070
MAFLA Onshore       15  0   1.05  0.150
Mid-West            16  0   1.05  0.150
Appalachia          17  0   1.05  0.120
North Alaska        20  0   1.05  0.150
MacKenzie Delta     21  0   1.05  0.150
Alberta             18  0   1.13  0.190
British Columbia    19  0   1.13  0.150
Sable Island         0  0   0.00  0.000
Distrigas            0  0   0.00  0.000
Cove Point           0  0   0.00  0.000
Elba Island          0  0   0.00  0.000
Lake Charles         0  0   0.00  0.000
Arctic Islands       0  0   0.00  0.000
Atlantic Offshore   22  0   1.05  0.150
Mexico-Supply       23  0   1.13  0.150
Alliance-Supply      0  0   0.00  0.000
```

Description of File: NODE.SPC

Data Element	Description	Format
1	Region name	a20
2	Supply region indicator	I2, 1X
3	Demand region indicator (read and used only in the DI model)	I2, 1X
4	Supply load factor	F6.3
5	Gathering Cost, $/Mcf	F6.3

Explanation of NODE.SPC

NODE.SPC is used to define which supply nodes and demand regions correspond to which region numbers. The last Data element is the gathering cost. It is subtracted away from the price to account for the cost of the processes between the wellhead and the pipeline.

Intended Uses of NODE.SPC

The only parameter in NODE.SPC that is likely to be changed is the gathering cost. Although this file could be used in the incorporation of a new region/node, such a change would involve altering the source code.

Table D-13

Data Input File: OTH_SUP.SPC (Location: \GSAM\DEMDINTG)
This file contains specifications for other supply projects.

```
ANGST               North Alaska        2010   1.50     0.00
MacKenzie Delta     MacKenzie Delta     2010   1.50     0.00
Arctic Islands      Arctic Islands      2015   2.00     0.00
Sable Island        Sable Island        2005   1.00   200.00
LNG(DG) - Current   Distrigas           1994   2.59    32.90
LNG(CP) - Current   Cove Point          1994   2.50     0.00
LNG(EI) - Current   Elba Island         1994   2.50     0.00
LNG(LC) - Current   Lake Charles        1994   1.71    17.90
LNG(DG) - $2.50 ThrsDistrigas           1993   2.50    50.90
LNG(CP) - $2.50 ThrsCove Point          1993   2.50     0.00
LNG(EI) - $2.50 ThrsElba Island         1993   2.50     0.00
LNG(LC) - $2.50 ThrsLake Charles        1993   2.50    10.00
LNG(DG) - $3.00 ThrsDistrigas           1993   3.00    20.00
LNG(CP) - $3.00 ThrsCove Point          1993   3.00     0.00
LNG(EI) - $3.00 ThrsElba Island         1993   3.00     0.00
LNG(LC) - $3.00 ThrsLake Charles        1993   3.00     0.00
LNG(DG) - $3.50 ThrsDistrigas           1993   3.50     0.00
LNG(CP) - $3.50 ThrsCove Point          1993   3.50     0.00
LNG(EI) - $3.50 ThrsElba Island         1993   3.50     0.00
LNG(LC) - $3.50 ThrsLake Charles        1993   3.50     0.00
```

Description of File: OTH_SUP.SPC

Data Element	Description	Format
1	Name of supply project	A20
2	Node name	A20
3	First year that project is allowed	I4, 1X
4	Levelized price of extra supply project ($/MCF)	F6.2, 1X
5	Maximum quantity that can flow at this price (MMCF/day)	F7.1

Explanation

These supply projects will be active when the current gas price is at or above the levelized price of the supply project.

Intended Uses

This file can be used to “force” a supply project to be activated or not activated, and can be used to create new supply projects or change the quantities for sensitivity analysis.

Table D-14

Input Data File: PFLAG.SPC (Location: \GSAM\DEMDINTG)
A “1” indicates that the intermediate output files are to be printed, and a “0” signals do not print. This file needs to be in both the Exploration and Production Module and the Demand and Integrating Modules.

```
      ←———  Print E&P outputs
1
1
      ←———  Print D&I outputs
```

Table D-15

Data Input File: REFMARG.SPC (Location: \GSAM\DEMDINTG)
This file contains data on refinery markups.

```
c*** Gulf Coast refinery margin in $/MCF (added to price of crude)
c***                 1994  1995  2000  2005  2010  2015  2020
distillate           0.64  0.64  0.64  0.64  0.64  0.64  0.64
1% sulfur resid     -1.31 -1.31 -1.31 -1.31 -1.31 -1.31 -1.31
3% sulfur resid     -1.54 -1.54 -1.54 -1.54 -1.54 -1.54 -1.54
```

Explanation

REFMARG.SPC shows the refinery margins over time. The price of a petroleum product is determined for two sectors, industrial and EU. It is calculated from the price of crude, the Gulf Coast refinery margin (given here), and the product-specific regional markup (due to transportation, given in DISTIND.SPC, or which ever product and sector is being analyzed).

Intended Uses

The refinery margins can change, affecting the demand for petroleum products in the industrial sector and the electric utilities sector.

Table D-16

Input Data File: RES_DEM.SPC (Location: \GSAM\DEMDINTG)
This file contains residential demand specifications by GSAM demand regions.

```
c** residential demand file  annual annual  annual  annual  annual
c**                 base    growth growth  growth  growth  growth
c**                 demd    rate % rate %  rate %  rate %  rate %
c**                 1995    [1996, (2000,  (2005,  (2010,  (2015,
c**                 (BCF)   2000]  2005]   2010]   2015]   2020]
"New England       "  173.7   0.411  0.325   0.787   0.943   0.930
"Middle Atlantic   "  831.6  -0.244 -0.326  -0.040   0.060   0.056
"South Atlantic    "  394.1   1.933  2.181   2.104   1.927   1.919
"Florida           "   14.5   1.988  2.155   2.049   1.953   1.950
"East South Central"  202.7   1.410  1.595   1.501   1.359   1.363
"East North Central" 1535.6  -0.468  0.063   0.171   0.263   0.264
"West South Central"  368.8   1.627  0.812   1.031   1.107   1.107
"West North Central"  480.8   0.359  0.681   0.766   0.859   0.863
"Mountain South    "   55.6   2.918  1.482   1.516   1.511   1.499
"Mountain North    "  215.1   2.921  1.493   1.516   1.501   1.499
"California        "  477.5   1.895  0.880   1.190   1.176   1.173
"Pacific Northwest "   80.1   1.899  0.871   1.190   1.182   1.171
"Canada-East       "  350.8   1.574  1.220   1.287   1.284   1.290
"Western Canada    "  164.9   1.161  0.911   1.008   1.010   1.009
"BC-Demand         "   70.2   1.654  1.601   1.642   1.643   1.635
"Mexico-Demand     "   27.0   3.714  6.791   7.565   7.563   7.573
```

Table D-17

Data Input File: GASPRC.STR (Location: \GSAM\DEMDINTG)
This file is a starting gas price file for the E&P Module in a fully integrated run. It is copied to GASPRC.NEW. It has been shortened to fit to one page in the Appendix. The actual file has years from 1993 through 2020. Price tracks 1 and 2 out of 5 total are shown.

```
Pacific Offshore      2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Pacific Onshore       2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
San Juan              2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Rockies Foreland      2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Williston             2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Permian               2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Mid-Continent         2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Arkla-East Texas      2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Texas Gulf Coast      2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Gulf of Mexico-West   2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Gulf of Mexico-Cntr   2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Norphlet              2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Gulf of Mexico-East   2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
So-Louisiana          2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
MAFLA Onshore         2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Mid-West              2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Appalachia            2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Alberta               2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
British Columbia      2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
North Alaska          2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
MacKenzie Delta       2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Atlantic Offshore     2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Mexico-Supply         2  1   0.100   0.100   0.100   0.100   0.100   0.100   0.100
Pacific Offshore      2  2   3.069   3.069   3.219   3.219   3.219   3.219   3.219
Pacific Onshore       2  2   3.069   3.069   3.219   3.219   3.219   3.219   3.219
San Juan              2  2   2.825   2.825   2.965   2.965   2.965   2.965   2.965
Rockies Foreland      2  2   2.859   2.859   3.009   3.009   3.009   3.009   3.009
Williston             2  2   2.795   2.795   2.942   2.942   2.942   2.942   2.942
Permian               2  2   2.771   2.771   2.906   2.906   2.906   2.906   2.906
Mid-Continent         2  2   2.886   2.886   2.996   2.996   2.996   2.996   2.996
Arkla-East Texas      2  2   2.960   2.960   3.064   3.064   3.064   3.064   3.064
Texas Gulf Coast      2  2   2.899   2.899   3.011   3.011   3.011   3.011   3.011
Gulf of Mexico-West   2  2   2.881   2.881   2.992   2.992   2.992   2.992   2.992
Gulf of Mexico-Cntr   2  2   2.993   2.993   3.117   3.117   3.117   3.117   3.117
Norphlet              2  2   3.028   3.028   3.154   3.154   3.154   3.154   3.154
Gulf of Mexico-East   2  2   0.200   0.200   0.200   0.200   0.200   0.200   0.200
So-Louisiana          2  2   3.020   3.020   3.146   3.146   3.146   3.146   3.146
MAFLA Onshore         2  2   3.057   3.057   3.183   3.183   3.183   3.183   3.183
Mid-West              2  2   3.150   3.150   3.293   3.293   3.293   3.293   3.293
Appalachia            2  2   3.206   3.206   3.344   3.344   3.344   3.344   3.344
Alberta               2  2   2.831   2.831   2.975   2.975   2.975   2.975   2.975
British Columbia      2  2   2.875   2.875   1.539   1.539   1.539   1.539   1.539
North Alaska          2  2   0.200   0.200   0.200   0.200   0.200   0.200   0.200
MacKenzie Delta       2  2   0.200   0.200   0.502   0.502   0.502   0.502   0.502
Atlantic Offshore     2  2   0.200   0.200   0.200   0.200   0.200   0.200   0.200
Mexico-Supply         2  2   2.849   2.849   2.958   2.958   2.958   2.958   2.958
```

Description of File: GASPRC.STR

Data Element	Description	Format
1	Supply region name	A20
2	Temporary index, not currently used	I3
3	Gas price track number	I3
4-10	Input supply price for time periods ($/Mcf)1X, F7.3	

Explanation

The starting gas price provides prices for the first pass of the E&P Module (the “supply side”) so that a supply curve can be generated and then fed back to the D&I Module to incorporate demand

Table D-18

INPUT File: BOILERS.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the industrial boilers in the module. It has been abridged to fit the width of this page

```
100001    0    WSCR    9 AZ          ARIZONA   04    4       COCHISE  003    3
100002    0    WSCR    9 AZ          ARIZONA   04    4       COCHISE  003    3
100003    0    WSCR    9 AZ          ARIZONA   04    4       COCHISE  003    3
100004    0    WSCR    9 AZ          ARIZONA   04    4       COCHISE  003    3
100005    0    WSCR    9 AZ          ARIZONA   04    4      COCONINO  005    5
100007    0    WSCR    9 AZ          ARIZONA   04    4      COCONINO  005    5
100008    0    WSCR    9 AZ          ARIZONA   04    4      COCONINO  005    5
100009    0    WSCR    9 AZ          ARIZONA   04    4      COCONINO  005    5
100013    0    WSCR    9 AZ          ARIZONA   04    4      COCONINO  005    5
100014    0    WSCR    9 AZ          ARIZONA   04    4      COCONINO  005    5
100015    0    WSCR    9 AZ          ARIZONA   04    4      COCONINO  005    5
100016    0    WSCR    9 AZ          ARIZONA   04    4      COCONINO  005    5
100017    0    WSCR    9 AZ          ARIZONA   04    4      COCONINO  005    5
100018    0    WSCR    9 AZ          ARIZONA   04    4      COCONINO  005    5
100019    0    WSCR    9 AZ          ARIZONA   04    4      COCONINO  005    5
100022    0    WSCR    9 AZ          ARIZONA   04    4          GILA  007    7
100023    0    WSCR    9 AZ          ARIZONA   04    4          GILA  007    7
100024    0    WSCR    9 AZ          ARIZONA   04    4          GILA  007    7
100025    0    WSCR    9 AZ          ARIZONA   04    4          GILA  007    7
100028    0    WSCR    9 AZ          ARIZONA   04    4      MARICOPA  013   13
100029    0    WSCR    9 AZ          ARIZONA   04    4      MARICOPA  013   13
100030    0     CNV   15 CA       CALIFORNIA   06    6      MONTEREY  053   53
100031    0     CNV   15 CA       CALIFORNIA   06    6      MONTEREY  053   53
100032    0     CNV   15 CA       CALIFORNIA   06    6      MONTEREY  053   53
100034    0     CNV   15 CA       CALIFORNIA   06    6      MONTEREY  053   53
100036    0     CNV   15 CA       CALIFORNIA   06    6      MONTEREY  053   53
100037    0     CNV   15 CA       CALIFORNIA   06    6      MONTEREY  053   53
100038    0     CNV   15 CA       CALIFORNIA   06    6      MONTEREY  053   53
100039    0     CNV   15 CA       CALIFORNIA   06    6      MONTEREY  053   53
100040    0     CNV   15 CA       CALIFORNIA   06    6      MONTEREY  053   53
.
.
.
```

(continues with other boilers)

Description of File: BOILERS.SPC

Data Element	Description	Format
1	ORIS Code	I10
2	Coal demand region code	I5
3	NERC Region	A9
4	Region Code	I5
5	State Abbreviation	A2
6	State Name	A20
7	State Code	A5
8	State Code No	I5
9	County Name	A20
10	County Code	A5
11	County Code No	I5
12	City Name	A30
13	City Code	A15
14	Industrial Plant Numerical Identifier	A12
15	Individual Boiler Identifier	A12
16	Boiler efficiency (Steam Btu out / Fuel Btu in)	F10.2
17	Standard Classification Code (identifies boiler type, fuel type, end use)	A10
18	SIC code (2 digit)	A5
19	SIC name	A20
20	NOx emission rate (lb./MMbtu)	F10.4
21	SO2 emission rate (lb./MMbtu)	F20.8
22	CO2 emission rate (lb./MMbtu)	F20.8
23	Major fuel type (GAS, OIL, etc)	A10
24	Actual subtype (resid, distillate, etc.)	A10
25	Boiler Firing Capacity (MMBtu/hr)	F15.2
26	Steam Capacity (MMBtu/hr)	F15.5
27	Capacity Factor(percent of the year)	F25.8
28	Steam Demand (Tbtu)	F25.8
29	Annual NOx Emission in (tons per year)	F20.4
30	Annual CO2 Emission in (tons per year	F20.4
31	Flag for fuel type group	A15

Table D-19

INPUT File: CANBOIL.SPC (Location: \GSAM\DEMDINTG)
This file contains the Steam Demand and the Capacity Factor of the Canadian and Mexican boilers by group.

```
**************************************************
Canada and Mexico Industrial Boiler Data by Group
**************************************************
  Group 1: small boiler, burns gas or distillate
  Group 2: small boiler, burns gas or resid
  Group 3: small boiler, burns gas only
  Group 4: large boiler, burns gas or distillate
  Group 5: large boiler, burns gas or resid
  Group 6: large boiler, burns gas only

                              1995       1995
                              Steam      Capacity
                              Demand     Factor
Region                Group   (Tbtu)     (frac)
********************  *****   ********   ********
"Canada-East       "    1     172.600     0.500
"Canada-East       "    2     172.600     0.500
"Canada-East       "    3     172.600     0.500
"Canada-East       "    4       0.000     0.500
"Canada-East       "    5       0.000     0.500
"Canada-East       "    6       0.000     0.500
"Western Canada    "    1     138.300     0.500
"Western Canada    "    2     138.300     0.500
"Western Canada    "    3     138.300     0.500
"Western Canada    "    4       0.000     0.500
"Western Canada    "    5       0.000     0.500
"Western Canada    "    6       0.000     0.500
"BC-Demand         "    1      33.000     0.500
"BC-Demand         "    2      33.000     0.500
"BC-Demand         "    3      33.000     0.500
"BC-Demand         "    4       0.000     0.500
"BC-Demand         "    5       0.000     0.500
"BC-Demand         "    6       0.000     0.500
"Mexico-Demand     "    1       8.670     0.500
"Mexico-Demand     "    2       8.670     0.500
"Mexico-Demand     "    3       8.670     0.500
"Mexico-Demand     "    4       0.000     0.500
"Mexico-Demand     "    5       0.000     0.500
"Mexico-Demand     "    6       0.000     0.500
```

Table D-20

INPUT File: CANNUGS.SPC (Location: \GSAM\DEMDINTG)
This file contains the Steam Demand and the Capacity Factor of the Canadian and Mexican NUG boilers by group

```
*************************************************
Canada and Mexico Industrial Boiler Data by Group
*************************************************
  Group 1: used in IPM, burns gas or distillate
  Group 2: used in IPM, burns gas or resid
  Group 3: not used in IPM, burns gas or distillate
  Group 4: not used in IPM, burns gas or resid

                              1995       1995
                              Steam      Capacity
                              Demand     Factor
Region                Group   (Tbtu)     (frac)
*******************   *****   ********   ********
"Canada-East       "    1       0.00      0.500
"Canada-East       "    2       0.00      0.500
"Canada-East       "    3       0.00      0.500
"Canada-East       "    4       0.00      0.500
"Western Canada    "    1       0.00      0.500
"Western Canada    "    2       0.00      0.500
"Western Canada    "    3       0.00      0.500
"Western Canada    "    4       0.00      0.500
"BC-Demand         "    1       0.00      0.500
"BC-Demand         "    2       0.00      0.500
"BC-Demand         "    3       0.00      0.500
"BC-Demand         "    4       0.00      0.500
"Mexico-Demand     "    1       0.00      0.500
"Mexico-Demand     "    2       0.00      0.500
"Mexico-Demand     "    3       0.00      0.500
"Mexico-Demand     "    4       0.00      0.500
```

Table D-21

INPUT File: CAPS.SPC (Location: \GSAM\DEMDINTG)
This file contains the New (not cumulative) node capacity data.

```
c**************************************************************************
Forced NEW (not cumulative) capacity file (max,min,fixed,free)
Format (free format):
        1.Link# as in link_nde  2.Bound (UP,LO,FX,FR)
3.Start Year  4.Capacity Volume (MMcf/d)(real number)
c**************************************************************************
63 LO 1999  700.0 !Foothills (Alb->WNC)
24 LO 1999  700.0 !Northern Border (WNC->ENC)

46 LO 1999  150.0 !Distrigas (DG->NE)

79 LO 2000 1500.0 !Alliance (AS->ENC)

48 LO 2000  360.0 !Maritimes/Northeast (SI->NE)

45 LO 1999  178.0 !PNGTS (CANE->NE)

15 LO 2000 1000.0 !Destin (Norphlet->MAFLA)

8  LO 2000 200.0 !PNW->California

17 LO 2000 400.0 !Texas Gulf Coast ->So. Louisiana

37 LO 2002 500.0 !East North Central ->Mid-Atlantic

20 LO 2000 500.0 !Alberta ->East North Central
```

Dictionary:

UP=Upper Bound
LO=Lower Bound
FX=Fixed Bound
FR=Free (no bound)

Table D-22

INPUT File: COM_EFF.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the commercial energy efficiency by GSAM Demand region.

```
c** commercial energy efficiency file
c**                  base    growth growth  growth  growth  growth
c**                  eff     rate % rate %  rate %  rate %  rate %
c**                  1995    1995-  2000-   2005-   2010-   2015-
c**                  (frac)  2000   2005    2010    2015    2020
"New England       " 1.000  -0.100 -0.100  -0.100  -0.100  -0.100
"Middle Atlantic   " 1.000  -0.100 -0.100  -0.100  -0.100  -0.100
"South Atlantic    " 1.000  -0.100 -0.100  -0.100  -0.100  -0.100
"Florida           " 1.000  -0.100 -0.100  -0.100  -0.100  -0.100
"East South Central" 1.000  -0.100 -0.100  -0.100  -0.100  -0.100
"East North Central" 1.000  -0.100 -0.100  -0.100  -0.100  -0.100
"West South Central" 1.000  -0.100 -0.100  -0.100  -0.100  -0.100
"West North Central" 1.000  -0.100 -0.100  -0.100  -0.100  -0.100
"Mountain South    " 1.000  -0.100 -0.100  -0.100  -0.100  -0.100
"Mountain North    " 1.000  -0.100 -0.100  -0.100  -0.100  -0.100
"California        " 1.000  -0.100 -0.100  -0.100  -0.100  -0.100
"Pacific Northwest " 1.000  -0.100 -0.100  -0.100  -0.100  -0.100
"Canada-East       " 1.000  -0.500 -0.500  -0.500  -1.000  -1.000
"Western Canada    " 1.000  -1.000 -1.000  -1.000  -1.000  -1.000
"BC-Demand         " 1.000  -1.000 -1.000  -1.000  -1.000  -1.000
"Mexico-Demand     " 1.000  -1.000 -1.000  -1.000  -1.000  -1.000
```

Table D-23

INPUT File: COM_ELS.SPC (Location: \GSAM\DEMDINTG)
This file contains the commercial demand elasticities by GSAM demand region.

```
c** commercial demand elasticities file
c**                  price   GNP     energy
c**
"New England       " -0.25   0.82    1.00
"Middle Atlantic   " -0.25   0.82    1.00
"South Atlantic    " -0.25   0.82    1.00
"Florida           " -0.25   0.82    1.00
"East South Central" -0.25   0.82    1.00
"East North Central" -0.25   0.82    1.00
"West South Central" -0.25   0.82    1.00
"West North Central" -0.25   0.82    1.00
"Mountain South    " -0.25   0.82    1.00
"Mountain North    " -0.25   0.82    1.00
"California        " -0.25   0.82    1.00
"Pacific Northwest " -0.25   0.82    1.00
"Canada-East       " -0.25   0.82    1.00
"Western Canada    " -0.25   0.82    1.00
"BC-Demand         " -0.25   0.82    1.00
"Mexico-Demand     " -0.25   0.82    1.00
```

Table D-24

INPUT File: COM_GNP.SPC (Location: \GSAM\DEMDINTG)
This file contains commercial sector GNP values and growth rates by GSAM demand region.

```
c** commercial GNP values    annual annual  annual  annual  annu
c** sector          base     growth growth  growth  growth  grow
c**                 gnp      rate % rate %  rate %  rate %  rate
c**                 1995     [1996, (2000,  (2005,  (2010,  (2015,
c**                 (MM$)    2000]  2005]   2010]   2015]   2020]
"New England       "  413.8   1.281  1.789   1.578   1.349   1.165
"Middle Atlantic   " 1168.7   0.730  1.479   1.300   1.105   0.959
"South Atlantic    "  929.1   0.901  2.024   1.799   1.543   1.336
"Florida           "  338.0   2.011  2.561   2.159   1.763   1.458
"East South Central"  377.1   1.710  1.927   1.704   1.446   1.246
"East North Central" 1189.7   1.172  1.702   1.499   1.279   1.106
"West South Central"  761.7   1.436  1.935   1.708   1.459   1.261
"West North Central"  484.7   1.597  1.841   1.594   1.327   1.136
"Mountain South    "  147.1   0.860  2.541   2.149   1.785   1.491
"Mountain North    "  265.1   2.033  2.498   2.150   1.802   1.512
"California        "  919.3   1.972  2.316   1.992   1.652   1.406
"Pacific Northwest "  229.7   1.808  2.256   1.954   1.644   1.408
"Canada-East       "  382.2   2.428  1.804   1.639   1.599   1.600
"Western Canada    "   88.8   2.991  2.426   1.764   2.680   2.199
"BC-Demand         "   71.4   2.757  2.199   2.002   2.003   2.004
"Mexico-Demand     "  100.0   2.493  3.511   3.497   3.505   3.492
```

Table D-25

INPUT File: COM_LD.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the seasonal commercial load factors by GSAM demand region.

```
c** commercial load factors by season
c**                      season    season   season   season
c**                      1         2        3        4
"New England       "     2.46      1.77     1.53     0.70
"Middle Atlantic   "     2.20      1.87     1.59     0.67
"South Atlantic    "     2.42      1.79     1.58     0.68
"Florida           "     1.80      1.24     1.17     0.90
"East South Central"     2.62      1.93     1.65     0.63
"East North Central"     2.91      1.88     1.73     0.60
"West South Central"     2.14      1.49     1.40     0.78
"West North Central"     2.51      1.86     1.68     0.63
"Mountain South    "     2.04      1.49     1.37     0.79
"Mountain North    "     2.28      1.73     1.59     0.68
"California        "     1.76      1.29     1.17     0.89
"Pacific Northwest "     2.34      1.62     1.57     0.70
"Canada-East       "     2.53      1.91     1.71     0.61
"Western Canada    "     2.84      1.68     1.60     0.67
"BC-Demand         "     2.58      1.61     1.52     0.71
"Mexico-Demand     "     2.14      1.49     1.40     0.78
```

Table D-26

INPUT File: COM_PRC.SPC (Location: \GSAM\DEMDINTG)
This file contains commercial gas prices and growth rates by GSAM demand region.

```
c** commercial prices file  annual annual  annual  annual  annu
c**                  base    growth growth  growth  growth  grow
c**                  price   rate % rate %  rate %  rate %  rate
c**                  1995    [1996, (2000,  (2005,  (2010,  (2015,
c**                  ($/MCF) 2000]  2005]   2010]   2015]   2020]
"New England       "  6.98    0.398 -1.357  -1.423  -1.022  -1.006
"Middle Atlantic   "  5.97    2.758 -0.772  -0.961  -0.745  -0.773
"South Atlantic    "  5.42    1.883 -1.382  -1.370  -1.227  -1.220
"Florida           "  5.41    1.921 -1.382  -1.370  -1.227  -1.220
"East South Central"  4.95    3.783 -1.098  -1.050  -0.991  -1.002
"East North Central"  4.84    2.664 -0.439  -0.752  -0.663  -0.685
"West South Central"  4.26    1.012 -0.679  -0.655  -0.336  -0.342
"West North Central"  4.05    4.636 -0.517  -0.863  -1.034  -0.997
"Mountain South    "  4.84    0.813 -1.345  -1.352  -0.832  -0.819
"Mountain North    "  3.09    8.980 -0.640  -0.975  -1.025  -0.981
"California        "  5.71    6.409 -1.345  -1.471  -0.538  -0.522
"Pacific Northwest "  3.19   10.985 -0.567  -0.623  -0.602  -0.579
"Canada-East       "  9.99  -17.752  1.497   2.175   2.125   2.178
"Western Canada    "  3.08   -5.106  2.181   2.997   2.899   2.989
"BC-Demand         "  2.44    7.969  2.192   1.127   1.158   1.094
"Mexico-Demand     "  4.24    1.107 -0.679  -0.655  -0.336  -0.342
```

Table D-27

INPUT File: CRUDE.SPC (Location: \GSAM\DEMDINTG)
This file contains history and forecasts for crude oil prices.

```
Crude Oil Prices (95$/Bbl)
1993 17.23
1994 17.23
1995 17.23
1996 20.07
1997 19.73
1998 13.36
1999 13.67
2000 13.99
2001 14.31
2002 14.62
2003 14.95
2004 15.26
2005 15.58
2006 15.90
2007 16.22
2008 16.53
2009 16.86
2010 17.17
2011 17.17
2012 17.17
2013 17.17
2014 17.17
2015 17.17
2016 17.17
2017 17.17
2018 17.17
2019 17.17
2020 17.17
2021 17.17
2022 17.17
2023 17.17
2024 17.17
2025 17.17
```

Table D-28

INPUT File: DMNCRV.SPC (Location: \GSAM\DEMDINTG)
This file contains the step function approximations of the sectoral demand curves.

```
Residential Demand Curve
Step function approximation of the residential demand curve
Price Points
1.00                                      ! price 1 of demand curve
8.70                                      ! price n of demand curve
0.07                                      ! step of demand curve
4                                         ! # of fixed prices before price 1
 0.10  0.25  0.50  0.75  0.00  0.00  0.00 ! fixed prices before price 1
5                                         ! # of fixed prices after price n
10.00 15.00 25.00 50.00 99.00  0.00  0.00 ! fixed prices after price n

Commercial Demand Curve
Step function approximation of the commercial demand curve
Price Points
1.00                                      ! price 1 of demand curve
8.70                                      ! price n of demand curve
0.07                                      ! step of demand curve
4                                         ! # of fixed prices before price 1
 0.10  0.25  0.50  0.75  0.00  0.00  0.00 ! fixed prices before price 1
5                                         ! # of fixed prices after price n
10.00 15.00 25.00 50.00 99.00  0.00  0.00 ! fixed prices after price n

Industrial Boiler Demand Curve
Step function approximation of the commercial demand curve
Price Points
1.00                                      ! price 1 of demand curve
8.70                                      ! price n of demand curve
0.07                                      ! step of demand curve
4                                         ! # of fixed prices before price 1
 0.10  0.25  0.50  0.75  0.00  0.00  0.00 ! fixed prices before price 1
5                                         ! # of fixed prices after price n
10.00 15.00 25.00 50.00 99.00  0.00  0.00 ! fixed prices after price n

Industrial NUGS Demand Curve
Step function approximation of the commercial demand curve
Price Points
1.00                                      ! price 1 of demand curve
8.70                                      ! price n of demand curve
0.07                                      ! step of demand curve
4                                         ! # of fixed prices before price 1
 0.10  0.25  0.50  0.75  0.00  0.00  0.00 ! fixed prices before price 1
5                                         ! # of fixed prices after price n
10.00 15.00 25.00 50.00 99.00  0.00  0.00 ! fixed prices after price n
```

Table D-29

INPUT File: DUAL_PRC.SPC (Location: \GSAM\DEMDINTG)
This file contains the dual prices by node, year and load season.

Node	Year number	Load Season	Dual Price
1	6	1	34.1021
2	6	1	32.3894
3	6	1	31.5849
4	6	1	31.8153
5	6	1	31.1435
6	6	1	31.7803
7	6	1	29.7585
8	6	1	29.7545
9	6	1	27.6764
10	6	1	26.9087
11	6	1	29.1175
12	6	1	28.1773
13	6	1	31.8033
14	6	1	27.4985
15	6	1	27.9049
16	6	1	29.7258
17	6	1	28.5341
18	6	1	28.5341
19	6	1	26.6824
20	6	1	26.2786
21	6	1	25.8421
22	6	1	29.1593
23	6	1	28.8259
24	6	1	29.2159
25	6	1	29.1133
26	6	1	28.9348
27	6	1	30.1411
28	6	1	0.000000
29	6	1	31.1781
30	6	1	30.3698
31	6	1	30.5487
32	6	1	31.4437
33	6	1	31.5956
34	6	1	0.000000
35	6	1	0.000000
36	6	1	26.9266
37	6	1	27.0438
38	6	1	0.000000

Table D-30

INPUT File: DUAL_PRC.STR (Location: \GSAM\DEMDINTG)
This file contains user-specified starting dual prices by node, year and load season.

```
Node  Year number         Load  Dual Price
                         Season

  1         1         1     5.0
  2         1         1     5.0
  3         1         1     5.0
  4         1         1     5.0
  5         1         1     5.0
  6         1         1     5.0
  7         1         1     5.0
  8         1         1     5.0
  9         1         1     5.0
 10         1         1     5.0
 11         1         1     5.0
 12         1         1     5.0
 13         1         1     5.0
 14         1         1     5.0
 15         1         1     5.0
 16         1         1     5.0
 17         1         1     5.0
 18         1         1     5.0
 19         1         1     5.0
 20         1         1     5.0
 21         1         1     5.0
 22         1         1     5.0
 23         1         1     5.0
 24         1         1     5.0
 25         1         1     5.0
 26         1         1     5.0
 27         1         1     5.0
 28         1         1     5.0
 29         1         1     5.0
 30         1         1     5.0
 31         1         1     5.0
 32         1         1     5.0
 33         1         1     5.0
 34         1         1     5.0
 35         1         1     5.0
 36         1         1     5.0
 37         1         1     5.0
 38         1         1     5.0
```

Table D-31

INPUT File: EFF.SPC (Location: \GSAM\DEMDINTG)
This file contains base energy efficiency by GSAM demand region. It adjusts the specifications for the Residential and Commercial sectors in the files RES_EFF.SPC and COM_EFF.SPC

```
c** Base Energy Efficiency by Demand Region
c**                           annual  annual  annual  annual  annu
c**                           growth  growth  growth  growth  grow
c**                           rate %  rate %  rate %  rate %  rate
c**                  base     [1996,  (2000,  (2005,  (2010,  (2015,
c**                  1995     2000]   2005]   2010]   2015]   2020]
c** Energy Efficiency (Btu/GDP$) **********************************
"New England       "  1.0000  0.000   0.000   0.000   0.000   0.000
"Middle Atlantic   "  1.0000  0.000   0.000   0.000   0.000   0.000
"South Atlantic    "  1.0000  0.000   0.000   0.000   0.000   0.000
"Florida           "  1.0000  0.000   0.000   0.000   0.000   0.000
"East South Central"  1.0000  0.000   0.000   0.000   0.000   0.000
"East North Central"  1.0000  0.000   0.000   0.000   0.000   0.000
"West South Central"  1.0000  0.000   0.000   0.000   0.000   0.000
"West North Central"  1.0000  0.000   0.000   0.000   0.000   0.000
"Mountain South    "  1.0000  0.000   0.000   0.000   0.000   0.000
"Mountain North    "  1.0000  0.000   0.000   0.000   0.000   0.000
"California        "  1.0000  0.000   0.000   0.000   0.000   0.000
"Pacific Northwest "  1.0000  0.000   0.000   0.000   0.000   0.000
"Canada-East       "  1.0000  0.000   0.000   0.000   0.000   0.000
"Western Canada    "  1.0000  0.000   0.000   0.000   0.000   0.000
"BC-Demand         "  1.0000  0.000   0.000   0.000   0.000   0.000
"Mexico-Demand     "  1.0000  0.000   0.000   0.000   0.000   0.000
```

Table D-32

INPUT File: EU1_LD.SPC (same format as EU2_LD.SPC, EU3_LD.SPC, EU4_LD.SPC), (Location: \GSAM\DEMDINTG)

The files of the kind EU#_LD.SPC (where “#” denotes the electric utility season and ranges from 1 to 4) contain load factors by electric utility seasons, gas seasons. and GSAM demand regions. EU1_LD.SPC is shown below. It has data for electric utility season 1.

```
GSAM region      Load factor   1       2       3      4
                  in gas
                  season:

"New England       "     0.00  0.00  0.00  1.50
"Middle Atlantic   "     0.00  0.00  0.00  1.50
"South Atlantic    "     0.00  0.00  0.00  1.50
"Florida           "     0.00  0.00  0.00  1.50
"East South Central"     0.00  0.00  0.00  1.50
"East North Central"     0.00  0.00  0.00  1.50
"West South Central"     0.00  0.00  0.00  1.50
"West North Central"     0.00  0.00  0.00  1.50
"Mountain South    "     0.00  0.00  0.00  1.50
"Mountain North    "     0.00  0.00  0.00  1.50
"California        "     0.00  0.00  0.00  1.50
"Pacific Northwest "     0.00  0.00  0.00  1.50
"Canada-East       "     0.00  0.00  0.00  1.50
"Western Canada    "     0.00  0.00  0.00  1.50
"BC-Demand         "     0.00  0.00  0.00  1.50
"Mexico-Demand     "     0.00  0.00  0.00  1.50
```

Table D-33

INPUT File: FEED.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the industrial feedstock demand for natural gas.

```
**********************************************************************
Industrial Feedstock Demand for gas(Bcf)
Data is from the 1998 Industrial Trends Analysis published by GRI.
The data for feed was based on total energy demand for feedstock
instead of energy demanded from natural gas for feedstock.
1995 numbers were derived from the national percentage of natural gas
for feed stock in 1995 multiplied by the regional demand
for energy for feedstock in 1995.
For GSAM regions that are subunits of Census Division, Demand was
weighted in 1995 by industrial GRP in 1995. Thus,
since Pacific Northwest made up 22 percent of the industrial GRP in
the Pacific it was given 22 percent of the feedstock
demand for natural gas.

**********************************************************************
                     base    growth growth  growth  growth  growth
                     eff     rate % rate %  rate %  rate %  rate %
                     1995    1995-  2000-   2005-   2010-   2015-
                             2000   2005    2010    2015    2020
"New England       "   6.9   2.037  1.334   1.005   0.721   0.721
"Middle Atlantic   "  74.3  -0.457  0.771   0.688   0.268   0.268
"South Atlantic    "  49.8   1.656  1.236   0.997   0.682   0.682
"Florida           "  12.9   1.656  1.236   0.997   0.682   0.682
"East South Central"  50.2   0.906  1.251   1.069   0.704   0.704
"East North Central" 102.2   0.570  1.116   0.911   0.555   0.555
"West South Central" 293.6   2.582  2.008   1.622   1.310   1.310
"West North Central"  43.6   1.363  0.974   0.762  -5.191  -5.191
"Mountain South    "   7.6   0.599  0.772   0.560   0.364   0.364
"Mountain North    "  13.2   0.599  0.772   0.560   0.364   0.364
"California        "  29.9   2.208  1.482   1.163   0.933   0.933
"Pacific Northwest "   8.5   2.208  1.482   1.163   0.933   0.933
"Canada-East       "   0.0   0.000  0.000   0.000   0.000   0.000
"Western Canada    "   0.0   0.000  0.000   0.000   0.000   0.000
"BC-Demand         "   0.0   0.000  0.000   0.000   0.000   0.000
"Mexico-Demand     "   0.0   0.000  0.000   0.000   0.000   0.000
```

Table D-34

INPUT File: FLOWS.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the forced pipeline flows among the nodes of the module.

```
c***********************************************************************
Forced pipeline flows file (minimum, maximum, equality)
Format (free format):
        1.Link# as in link_nde  2.Direction (F,R) 3.Bound (UP,LO,EQ)
4.Start Year  5.Season (1,2)  6.Flow Volume (MMcf/d)(real number)
c***********************************************************************
 1 F UP 1993 1    0.0 !ESC->APPL (season 1)
 1 F UP 1993 2    0.0 !ESC->APPL (season 2)
 1 F UP 1993 3    0.0 !ESC->APPL (season 3)
 1 F UP 1993 4    0.0 !ESC->APPL (season 4)

19 R UP 1993 1    0.0 !ENC->APPL (season 1)
19 R UP 1993 2    0.0 !ENC->APPL (season 2)
19 R UP 1993 3    0.0 !ENC->APPL (season 3)
19 R UP 1993 4    0.0 !ENC->APPL (season 4)

35 R UP 1993 1    0.0 !MA->APPL (season 1)
35 R UP 1993 2    0.0 !MA->APPL (season 2)
35 R UP 1993 3    0.0 !MA->APPL (season 3)
35 R UP 1993 4    0.0 !MA->APPL (season 4)

36 R UP 1993 1    0.0 !MA->CANE (season 1)
36 R UP 1993 2    0.0 !MA->CANE (season 2)
36 R UP 1993 3    0.0 !MA->CANE (season 2)
36 R UP 1993 4    0.0 !MA->CANE (season 2)

45 R UP 1993 1    0.0 !NE->CANE (season 1)
45 R UP 1993 2    0.0 !NE->CANE (season 2)
45 R UP 1993 3    0.0 !NE->CANE (season 3)
45 R UP 1993 4    0.0 !NE->CANE (season 4)

51 R UP 1993 1    0.0 !SA->APPL (season 1)
51 R UP 1993 2    0.0 !SA->APPL (season 2)
51 R UP 1993 3    0.0 !SA->APPL (season 3)
51 R UP 1993 4    0.0 !SA->APPL (season 4)

63 R UP 1993 1    0.0 !WNC->ALB (season 1)
63 R UP 1993 2    0.0 !WNC->ALB (season 2)
63 R UP 1993 3    0.0 !WNC->ALB (season 3)
63 R UP 1993 4    0.0 !WNC->ALB (season 4)

74 R UP 1993 1    0.0 !Alliance->BC (season 1)
74 R UP 1993 2    0.0 !Alliance->BC (season 2)
74 R UP 1993 3    0.0 !Alliance->BC (season 3)
74 R UP 1993 4    0.0 !Alliance->BC (season 4)

75 R UP 1993 1    0.0 !Alliance->ALB (season 1)
75 R UP 1993 2    0.0 !Alliance->ALB (season 2)
75 R UP 1993 3    0.0 !Alliance->ALB (season 3)
75 R UP 1993 4    0.0 !Alliance->ALB (season 4)
```

Dictionary:
F=Formal Direction
R=Reverse Direction
UP=Upper Bound
LO=Lower Bound
EQ=Equality

Table D-35

INPUT File: GASPRC.HIS (Location: \GSAM\DEMDINTG)
This file contains historical gas prices from 1993 to 1997 by GSAM supply region and price track (tracks go from 1 to 5; 1 and 2 are shown here).

```
c        historical prices
c                              1993    1994    1995    1996    1997
Pacific Offshore      2  1   1.920   1.835   1.749   1.706   1.908
Pacific Onshore       2  1   1.920   1.835   1.749   1.696   1.908
San Juan              2  1   1.740   1.655   1.569   1.520   1.711
Rockies Foreland      2  1   1.730   1.610   1.489   1.519   1.711
Williston             2  1   1.650   1.535   1.419   1.450   1.632
Permian               2  1   1.810   1.720   1.629   1.588   1.770
Mid-Continent         2  1   1.740   1.660   1.579   1.529   1.721
Arkla-East Texas      2  1   1.850   1.760   1.669   1.617   1.809
Texas Gulf Coast      2  1   1.780   1.695   1.609   1.558   1.750
Gulf of Mexico-West   2  1   1.780   1.695   1.609   1.558   1.750
Gulf of Mexico-Cntr   2  1   1.820   1.740   1.659   1.608   1.800
Norphlet              2  1   1.840   1.755   1.669   1.627   1.829
Gulf of Mexico-East   2  1   2.130   2.140   2.149   2.755   3.122
So-Louisiana          2  1   1.850   1.765   1.679   1.637   1.829
MAFLA Onshore         2  1   1.800   1.715   1.629   1.588   1.790
Mid-West              2  1   1.870   1.785   1.699   1.648   1.849
Appalachia            2  1   1.960   1.880   1.799   1.755   1.987
Alberta               2  1   1.225   1.285   1.444   1.440   1.691
British Columbia      2  1   1.585   1.491   1.397   1.499   1.750
North Alaska          2  1   0.100   0.100   0.100   0.100   0.100
MacKenzie Delta       2  1   0.954   0.912   0.869   0.958   0.962
Atlantic Offshore     2  1   0.990   0.990   0.989   0.958   0.962
Mexico-Supply         2  1   1.800   1.720   1.639   1.529   1.721
Pacific Offshore      2  2   1.920   1.835   1.749   1.706   1.908
Pacific Onshore       2  2   1.920   1.835   1.749   1.696   1.908
San Juan              2  2   1.740   1.655   1.569   1.520   1.711
Rockies Foreland      2  2   1.730   1.610   1.489   1.519   1.711
Williston             2  2   1.650   1.535   1.419   1.450   1.632
Permian               2  2   1.810   1.720   1.629   1.588   1.770
Mid-Continent         2  2   1.740   1.660   1.579   1.529   1.721
Arkla-East Texas      2  2   1.850   1.760   1.669   1.617   1.809
Texas Gulf Coast      2  2   1.780   1.695   1.609   1.558   1.750
Gulf of Mexico-West   2  2   1.780   1.695   1.609   1.558   1.750
Gulf of Mexico-Cntr   2  2   1.820   1.740   1.659   1.608   1.800
Norphlet              2  2   1.840   1.755   1.669   1.627   1.829
Gulf of Mexico-East   2  2   2.130   2.140   2.149   2.755   3.122
So-Louisiana          2  2   1.850   1.765   1.679   1.637   1.829
MAFLA Onshore         2  2   1.800   1.715   1.629   1.588   1.790
Mid-West              2  2   1.870   1.785   1.699   1.648   1.849
Appalachia            2  2   1.960   1.880   1.799   1.755   1.987
Alberta               2  2   1.225   1.285   1.444   1.440   1.691
.
.
```

(continues with other price tracks)

Table D-36

INPUT File: IND_EI.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the energy intensity of the industrial boilers and NUGs in the module.

```
**********************************************************************
Industrial BOILER Energy Intensity ($/Btu)
**********************************************************************
                    base    growth growth  growth  growth  growth
                    eff     rate % rate %  rate %  rate %  rate %
                    1995    1995-  2000-   2005-   2010-   2015-
                            2000   2005    2010    2015    2020
"New England       " 1.000   1.000 0.000   0.000   0.000   0.000
"Middle Atlantic   " 1.000   0.000 0.000   0.000   0.000   0.000
"South Atlantic    " 1.000   0.000 0.000   0.000   0.000   0.000
"Florida           " 1.000   0.000 0.000   0.000   0.000   0.000
"East South Central" 1.000   0.000 0.000   0.000   0.000   0.000
"East North Central" 1.000   0.000 0.000   0.000   0.000   0.000
"West South Central" 1.000   0.000 0.000   0.000   0.000   0.000
"West North Central" 1.000   0.000 0.000   0.000   0.000   0.000
"Mountain South    " 1.000   0.000 0.000   0.000   0.000   0.000
"Mountain North    " 1.000   0.000 0.000   0.000   0.000   0.000
"California        " 1.000   0.000 0.000   0.000   0.000   0.000
"Pacific Northwest " 1.000   0.000 0.000   0.000   0.000   0.000
"Canada-East       " 1.000   0.000 0.000   0.000   0.000   0.000
"Western Canada    " 1.000   0.000 0.000   0.000   0.000   0.000
"BC-Demand         " 1.000   0.000 0.000   0.000   0.000   0.000
"Mexico-Demand     " 1.000   0.000 0.000   0.000   0.000   0.000

**********************************************************************
Industrial NUGS Energy Intensity ($/Btu)
**********************************************************************
                    base    growth growth  growth  growth  growth
                    eff     rate % rate %  rate %  rate %  rate %
                    1995    1995-  2000-   2005-   2010-   2015-
                            2000   2005    2010    2015    2020
"New England       " 1.000   0.000 0.000   0.000   0.000   0.000
"Middle Atlantic   " 1.000   0.000 0.000   0.000   0.000   0.000
"South Atlantic    " 1.000   0.000 0.000   0.000   0.000   0.000
"Florida           " 1.000   0.000 0.000   0.000   0.000   0.000
"East South Central" 1.000   0.000 0.000   0.000   0.000   0.000
"East North Central" 1.000   0.000 0.000   0.000   0.000   0.000
"West South Central" 1.000   0.000 0.000   0.000   0.000   0.000
"West North Central" 1.000   0.000 0.000   0.000   0.000   0.000
"Mountain South    " 1.000   0.000 0.000   0.000   0.000   0.000
"Mountain North    " 1.000   0.000 0.000   0.000   0.000   0.000
"California        " 1.000   0.000 0.000   0.000   0.000   0.000
"Pacific Northwest " 1.000   0.000 0.000   0.000   0.000   0.000
"Canada-East       " 1.000   0.000 0.000   0.000   0.000   0.000
"Western Canada    " 1.000   0.000 0.000   0.000   0.000   0.000
"BC-Demand         " 1.000   0.000 0.000   0.000   0.000   0.000
"Mexico-Demand     " 1.000   0.000 0.000   0.000   0.000   0.000
```

Table D-37

INPUT File: IND_ELS.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the industrial elasticities of the different fuels by GSAM demand region.

```
*************************************************************************
GAS ONLY: Industrial Elasticities
*************************************************************************
                            energy    gas     alternate
                     gnp   intensity price   price
"New England       "  0.00    1.98   -0.98    0.00
"Middle Atlantic   "  0.00    1.98   -0.98    0.00
"South Atlantic    "  0.00    1.98   -0.98    0.00
"Florida           "  0.00    1.98   -0.98    0.00
"East South Central"  0.00    1.98   -0.98    0.00
"East North Central"  0.00    1.98   -0.98    0.00
"West South Central"  0.00    0.58   -0.81    0.00
"West North Central"  0.00    0.58   -0.81    0.00
"Mountain South    "  0.00    0.58   -0.81    0.00
"Mountain North    "  0.00    0.58   -0.81    0.00
"California        "  0.00    0.58   -0.81    0.00
"Pacific Northwest "  0.00    0.58   -0.81    0.00
"Canada-East       "  0.00    1.98   -0.98    0.00
"Western Canada    "  0.00    0.58   -0.81    0.00
"BC-Demand         "  0.00    0.58   -0.81    0.00
"Mexico-Demand     "  0.00    0.58   -0.81    0.00

*********************************************************
DISTILLATE: Industrial Elasticities
*********************************************************
                            energy    gas    distillate
                     gnp   intensity price  price
"New England       "  0.00    0.51   -0.91   0.13
"Middle Atlantic   "  0.00    0.51   -0.91   0.13
"South Atlantic    "  0.00    0.51   -0.91   0.13
"Florida           "  0.00    0.51   -0.91   0.13
"East South Central"  0.00    0.51   -0.91   0.13
"East North Central"  0.00    0.51   -0.91   0.13
"West South Central"  0.00    0.51   -0.91   0.13
"West North Central"  0.00    0.51   -0.91   0.13
"Mountain South    "  0.00    0.51   -0.91   0.13
"Mountain North    "  0.00    0.51   -0.91   0.13
"California        "  0.00    0.51   -0.91   0.13
"Pacific Northwest "  0.00    0.51   -0.91   0.13
"Canada-East       "  0.00    0.51   -0.91   0.13
"Western Canada    "  0.00    0.51   -0.91   0.13
"BC-Demand         "  0.00    0.51   -0.91   0.13
"Mexico-Demand     "  0.00    0.51   -0.91   0.13

*********************************************************
RESIDUAL: Industrial Elasticities
*********************************************************
                            energy    gas    residual
                     gnp   intensity price  price
"New England       "  0.00    2.01   -1.49   0.43
"Middle Atlantic   "  0.00    2.01   -1.49   0.43
"South Atlantic    "  0.00    2.01   -1.49   0.43
"Florida           "  0.00    2.01   -1.49   0.43
"East South Central"  0.00    2.01   -1.49   0.43
"East North Central"  0.00    2.01   -1.49   0.43
"West South Central"  0.00    0.57   -0.91   0.08
"West North Central"  0.00    0.57   -0.91   0.08
"Mountain South    "  0.00    0.57   -0.91   0.08
"Mountain North    "  0.00    0.57   -0.91   0.08
"California        "  0.00    0.57   -0.91   0.08
"Pacific Northwest "  0.00    0.57   -0.91   0.08
"Canada-East       "  0.00    2.01   -1.49   0.43
"Western Canada    "  0.00    0.57   -0.91   0.08
"BC-Demand         "  0.00    0.57   -0.91   0.08
"Mexico-Demand     "  0.00    0.57   -0.91   0.08
```

Table D-38

INPUT File: IND_LD.SPC (Location: \GSAM\DEMDINTG)
This file contains for the industrial load factors by season and GSAM demand region.

```
c** industrial load factors by season
c**                       season    season   season   season
c**                       1         2        3        4
"New England       "     1.03      1.03     1.04     0.98
"Middle Atlantic   "     1.19      1.19     1.12     0.93
"South Atlantic    "     0.94      0.94     0.98     1.01
"Florida           "     0.97      0.97     0.99     1.01
"East South Central"     1.08      1.08     1.07     0.96
"East North Central"     1.24      1.24     1.18     0.90
"West South Central"     1.02      1.02     1.01     0.99
"West North Central"     1.13      1.13     1.11     0.94
```

```
"Mountain South    "     1.07      1.07     1.01     0.98
"Mountain North    "     1.15      1.15     1.08     0.95
"California        "     0.93      0.93     0.95     1.02
"Pacific Northwest "     1.03      1.03     1.03     0.98
"Canada-East       "     1.28      1.28     1.20     0.89
"Western Canada    "     1.06      1.06     1.04     0.98
"BC-Demand         "     1.01      1.01     0.92     1.03
"Mexico-Demand     "     1.02      1.02     1.01     0.99
```

Table D-39

INPUT File: IND_PRC.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the industrial base gas prices by fuel and GSAM demand region.

```
**********************************************************************
Industrial Base Gas Prices (95$/Mcf), distillate & resid prices
**********************************************************************
                        base  base   base
                        gas   dist   resid
                        price price  price
"New England       "    4.36  5.24   2.89
"Middle Atlantic   "    3.98  5.22   3.09
"South Atlantic    "    3.31  4.72   2.84
"Florida           "    3.16  4.74   2.80
"East South Central"    3.04  4.88   2.58
"East North Central"    3.59  5.07   2.65
"West South Central"    2.16  0.30   0.02
"West North Central"    2.80  5.09   2.47
"Mountain South    "    3.12  5.19   2.60
"Mountain North    "    3.13  5.66   2.70
"California        "    3.77  5.60   2.79
"Pacific Northwest "    2.98  5.33   2.83
"Canada-East       "    4.36  5.24   2.89
"Western Canada    "    2.80  5.09   2.47
"BC-Demand         "    2.98  5.33   2.83
"Mexico-Demand     "    3.12  5.19   2.60
```

Table D-40

INPUT File: LNG.SPC (Location: \GSAM\DEMDINTG)
This file contains LNG capacities and costs for existing and new plants.

```
c* LNG Capacities and Costs
c*                  >>>>>>>>>>>>>EXISTING<<<<<<<<<<<<<<<<<<<<<  >>>>>>>>>>>>>NEW<<<<<<<<<<<<<<<<<<<<<<<<<<
c*                                            Annual  Var.   Fixed                       Annual  Var.   Fixed  LOW
c*                  Delivery Stor.   1st  Demand  O&M      O&M  Deliv. Stor.   1st   Demand  O&M    O&M    BND %
c*                   Capacity Cap.   year   Charge Charge Charge Cap.   Cap.    year  Charge  Charge Charge OLD,
Peak
c*   Demand Region   MMcfd   MMcf  avail. M$/MMcf $/Mcf   $/Mcf MMcfd   MMcf  Avail. M$/MMcf $/Mcf   $/Mcf
"New England       "  1400   18125   1990   54.98   2.09   0.00   400   4000   2000   61.09   1.15   0.00  10.0
"Middle Atlantic   "  2160   12511   1990   54.98   2.09   0.00   400   4000   2000   61.09   1.15   0.00  10.0
"South Atlantic    "  2657   19762   1990   47.81   2.09   0.00   400   4000   2000   53.13   1.15   0.00  10.0
"Florida           "     0       0   1990    0.00   0.00   0.00   400   4000   2000   53.13   1.15   0.00  10.0
"East South Central"   855    7035   1990   47.81   2.09   0.00   400   4000   2000   53.13   1.15   0.00  10.0
"East North Central"  1210    9950   1990   52.59   2.09   0.00   400   4000   2000   58.44   1.15   0.00  10.0
"West South Central"    57     188   1990   47.81   2.09   0.00   400   4000   2000   53.13   1.15   0.00  10.0
"West North Central"   815    8624   1990   52.59   2.09   0.00   400   4000   2000   58.44   1.15   0.00  10.0
"Mountain South    "     0       0   1990    0.00   0.00   0.00   400   4000   2000   53.13   1.15   0.00  10.0
"Mountain North    "   165    1825   1990   47.81   2.09   0.00   400   4000   2000   53.13   1.15   0.00  10.0
"California        "     0       0   1990    0.00   0.00   0.00   400   4000   2000   58.44   1.15   0.00  10.0
"Pacific Northwest "   300    2698   1990   52.59   2.09   0.00   400   4000   2000   58.44   1.15   0.00  10.0
"Canada-East       "   440    2330   1990   52.59   2.09   0.00   400   4000   2000   58.44   1.15   0.00  10.0
"Western Canada    "     0       0   1990    0.00   0.00   0.00   400   4000   2000   53.13   1.15   0.00  10.0
"BC-Demand         "   150     600   1990   47.81   2.09   0.00   400   4000   2000   53.13   1.15   0.00  10.0
"Mexico-Demand     "     0       0   1990    0.00   0.00   0.00   400   4000   2000   53.13   1.15   0.00  10.0
```

Table D-41

INPUT File: NOX.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the NOx allowance costs by fuel, GSAM demand region, start year, and gas season.

```
**************************************************************************
NOx Allowance Costs ($/Kwh) by Fuel by Region by Start Year by Gas Season
NOx also includes other non-SOX costs
***************************************************************************
Fuel/Region     Start Year Seas.1 Seas.2 Seas.3 Seas.4 Seas.5 Seas.6 Seas.7
***************************************************************************
"NUCLEAR"
"New England       "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Middle Atlantic   "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"South Atlantic    "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Florida           "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"East South Central"  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"East North Central"  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"West South Central"  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"West North Central"  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Mountain South    "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Mountain North    "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"California        "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Pacific Northwest "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Canada-East       "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Western Canada    "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"BC-Demand         "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Mexico-Demand     "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"COAL"
"New England       "  1993 0.0000 0.0000 0.0000 0.0035 0.0000 0.0000 0.0000
"Middle Atlantic   "  1993 0.0000 0.0000 0.0000 0.0035 0.0000 0.0000 0.0000
"South Atlantic    "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Florida           "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"East South Central"  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"East North Central"  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"West South Central"  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"West North Central"  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Mountain South    "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Mountain North    "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"California        "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Pacific Northwest "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Canada-East       "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Western Canada    "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"BC-Demand         "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Mexico-Demand     "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
```

.

.

.

(continues with other fuels)

Table D-42

INPUT File: NUGS.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the industrial NUGS in the module. It has been abridged to fit the width of the page

```
200001Y     COGEN        NT       SEV  CA      COMM         CT-GAS       GAS
200002Y     COGEN        NT       EXT  CA      COMM         CT-GAS       GAS
200003Y     COGEN        NT            CA      COMM         CT-GAS       GAS
200004Y     COGEN        NT       SER  CA      COMM         CT-GAS       GAS
200005Y     COGEN        NT       SER  CA      COMM         CT-GAS       GAS
200006Y     COGEN        NT            CA      COMM         CT-GAS       GAS
200007      COGEN        NT       EXT  CA      COMM         CT-GAS       GAS
200008Y     COGEN        NT       SER  CA      COMM         CT-GAS       GAS
200009Y     COGEN        NT            CA      COMM         CT-GAS       GAS
200010      COGEN        NT       SER  CA      COMM         CT-GAS       GAS
200011Y     COGEN        NT       EXT  CA      COMM         CT-GAS       GAS
200012Y     COGEN        NT       EXT  CA      COMM         CT-GAS       GAS
200014Y     COGEN        NT            CA       REF         CC-GAS       GAS
200015Y     COGEN        NT       EXT  CA       REF         CC-GAS       GAS
200016Y     COGEN        NT       SEV  CA      CHEM         CT-GAS       GAS
200017Y     COGEN        NT       SER  CA      CHEM         CT-GAS       GAS
200018      COGEN        NT       EXT  CA      PAPR         CT-GAS       GAS
200019Y     COGEN        NT       SER  CA       REF         CT-GAS       GAS
200020Y     COGEN        NT       SER  CA      CHEM         CT-GAS       GAS
200021Y     COGEN        NT       SEV  CA      PAPR         CT-GAS       GAS
200022      COGEN        NT       EXT  CA       REF         CT-GAS       GAS
200023Y     COGEN        NT       SER  CA       SCG         CT-GAS       GAS
200027Y     COGEN        NT       EXT  CA      MING         CC-GAS       GAS
200028Y     COGEN        NT            CA      FOOD         CC-GAS       GAS
200030Y     COGEN        NT       SER  CA      MING         CT-GAS       GAS
200031Y     COGEN        NT       EXT  CA      FOOD         CT-GAS       GAS
200032Y     COGEN        NT       SER  CA      MING         CT-GAS       GAS
200033Y     COGEN        NT       SER  CA      MING         CT-GAS       GAS
200034Y     COGEN        NT       EXT  CA      MING         CT-GAS       GAS
200035Y     COGEN        NT       SER  CA      MING         CT-GAS       GAS
200036Y     COGEN        NT       SER  CA      MING         CT-GAS       GAS
200037Y     COGEN        NT       SEV  CA       AGR         CT-GAS       GAS
200038Y     COGEN        NT       SER  CA      MING         CT-GAS       GAS
```

.

.

.

(continues with other NUGs)

Description of File: NUGS.SPC

Data Element	Description	Format
1	RID Code	I11
2	Type Flag (COGEN or not COGEN)	A1
3	“Traditional” or “Non-traditional” flag	A10
4	Nonattainment Designation	A10
5	State Abbreviation	A10
6	Edison Electric Institute SIC code (REF, CHEM, PAPR, etc.)	A2
7	IPM Primary Fuel/Generation Type	A10
8	IPM Primary Fuel Type Used	A15
9	Input firing rate MMBTU/hr.	A10
10	IPM generation capacity MW	A20
11	Start Year	F15.2
12	Estimated End Year	F15.2
13	Final NOx Rate lbs./MMBtu	I10
14	SOx Estimate lbs./MMBtu	I10
15	capacity factor	F15.5
16	Generation in Megawatt hours	F15.5
17	Estimated fuel usage in millions of BTU for Electricity	F15.5
18	Power/Heat (useful BTU of Electricity/useful BTU of Steam)	F15.2
19	millions of BTU of fuel used for Thermal	F15.2
20	millions of BTU of fuel used	F15.5
21	Heat rate in BTU per kWh	F15.1
22	Net Heat Rate in BTU per kWh ([Total fuel used – Fuel for steam]/kWh)	F15.1
23	SOx Tons Rate lbs./MMBtu	F10.1
24	NOx Tons Rate lbs./MMBtu	F10.1
25	CO2 Emissions Rate lbs./MMBtu	F15.5
26	CO2 Emissions in Tons mm Metric Tons	F15.5
27	Secondary Fuel	F15.5
28	EWG	F15.5
29	Flag for fuel type group	A15

Table D-43

OUTPUT File: ODUALS.SPC (Location: \GSAM\DEMDINTG)
This file contains the dual gas prices from each previous iteration of the model by GSAM demand region and year . It gets overwritten until the final iteration. Its initial condition is ODUALS.STR. The file has been abridged to fit the width of the page. The run years are specified in GEN_TML.SPC. In this file they are 1993 through 2020.

```
Pacific Offshore       0.000   0.000   0.000   0.000   0.000   8.653   8.620   2.484   2.579   2.601   2.825
Pacific Onshore        0.000   0.000   0.000   0.000   0.000   8.314   8.285   2.474   2.567   2.590   2.810
San Juan               0.000   0.000   0.000   0.000   0.000   7.717   7.691   2.249   2.336   2.357   2.564
Rockies Foreland       0.000   0.000   0.000   0.000   0.000   7.618   7.258   2.184   2.246   2.256   2.277
Williston              0.000   0.000   0.000   0.000   0.000   7.448   7.094   2.101   2.162   2.172   2.192
Permian                0.000   0.000   0.000   0.000   0.000   8.497   8.087   2.379   2.449   2.503   2.634
Mid-Continent          0.000   0.000   0.000   0.000   0.000   8.384   7.940   2.370   2.441   2.460   2.597
Arkla-East Texas       0.000   0.000   0.000   0.000   0.000   8.544   8.133   2.413   2.484   2.538   2.670
Texas Gulf Coast       0.000   0.000   0.000   0.000   0.000   8.471   8.061   2.353   2.423   2.477   2.608
Gulf of Mexico-West    0.000   0.000   0.000   0.000   0.000   8.513   8.099   2.348   2.418   2.472   2.603
Gulf of Mexico-Cntr    0.000   0.000   0.000   0.000   0.000   8.898   8.455   2.474   2.547   2.608   2.745
Norphlet               0.000   0.000   0.000   0.000   0.000   1.000   1.000   2.511   2.586   2.648   2.788
Gulf of Mexico-East    0.000   0.000   0.000   0.000   0.000   9.171   8.735   2.665   2.727   2.796   2.932
So-Louisiana           0.000   0.000   0.000   0.000   0.000   8.890   8.451   2.506   2.580   2.640   2.778
MAFLA Onshore          0.000   0.000   0.000   0.000   0.000   9.242   8.775   2.470   2.545   2.607   2.747
Mid-West               0.000   0.000   0.000   0.000   0.000   9.151   8.697   2.572   2.649   2.712   2.855
Appalachia             0.000   0.000   0.000   0.000   0.000   9.374   8.955   2.691   2.768   2.828   2.945
Alberta                0.000   0.000   0.000   0.000   0.000   8.091   8.189   2.366   2.441   2.456   2.576
British Columbia       0.000   0.000   0.000   0.000   0.000   8.177   8.275   2.406   2.470   2.476   2.578
North Alaska           0.000   0.000   0.000   0.000   0.000   1.000   1.000   1.610   1.641   1.655   1.709
MacKenzie Delta        0.000   0.000   0.000   0.000   0.000   1.000   1.000   2.546   2.621   2.490   2.610
Atlantic Offshore      0.000   0.000   0.000   0.000   0.000   1.000   1.000   1.000   1.000   1.000   1.000
Mexico-Supply          0.000   0.000   0.000   0.000   0.000   8.508   8.093   2.304   2.375   2.429   2.560
```

Table D-44

OUTPUT File: ODUALS.STR (Location: \GSAM\DEMDINTG)
This file contains user-specified starting dual gas prices by GSAM demand region and year. The file has been abridged to fit the width of the page. The run years are specified in GEN_TML.SPC. In this file they are 1993 through 2010.

```
Pacific Offshore       0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Pacific Onshore        0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
San Juan               0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Rockies Foreland       0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Williston              0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Permian                0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Mid-Continent          0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Arkla-East Texas       0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Texas Gulf Coast       0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Gulf of Mexico-West    0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Gulf of Mexico-Cntr    0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Norphlet               0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Gulf of Mexico-East    0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
So-Louisiana           0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
MAFLA Onshore          0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Mid-West               0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Appalachia             0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Alberta                0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
British Columbia       0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
North Alaska           0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
MacKenzie Delta        0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Atlantic Offshore      0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
Mexico-Supply          0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000   0.000
```

Table D-45

INPUT File: POP_GRP.SPC (Location: \GSAM\DEMDINTG)
This file contains data for Gross Regional Product and Population by GSAM demand region, and for Total Electricity Demand (for all regions).

```
c** Gross Regional Product, Population, and Total Elec Demand Inputs
c**                           annual  annual  annual  annual  annu
c**                           growth  growth  growth  growth  grow
c**                           rate %  rate %  rate %  rate %  rate
c**                  base     [1996,  (2000,  (2005,  (2010,  (2015,
c**                  1995     2000]   2005]   2010]   2015]   2020]
c** Gross Regional Product (Bill $95) ******************************
"New England       "  413.83  1.281   1.789   1.578   1.349   1.165
"Middle Atlantic   " 1168.72  0.730   1.479   1.300   1.105   0.959
"South Atlantic    "  929.07  0.901   2.024   1.799   1.543   1.336
"Florida           "  337.95  2.011   2.561   2.159   1.763   1.458
"East South Central"  377.11  1.710   1.927   1.704   1.446   1.246
"East North Central" 1189.73  1.172   1.702   1.499   1.279   1.106
"West South Central"  761.71  1.436   1.935   1.708   1.459   1.261
"West North Central"  484.73  1.597   1.841   1.594   1.327   1.136
"Mountain South    "  147.13  0.860   2.541   2.149   1.785   1.491
"Mountain North    "  265.10  2.033   2.498   2.150   1.802   1.512
"California        "  919.30  1.972   2.316   1.992   1.652   1.406
"Pacific Northwest "  229.68  1.808   2.256   1.954   1.644   1.408
"Canada-East       "  382.20  2.428   1.804   1.639   1.599   1.600
"Western Canada    "   88.80  2.991   2.426   2.242   2.200   2.199
"BC-Demand         "   71.40  2.757   2.199   2.002   2.003   2.004
"Mexico-Demand     "  100.00  2.493   3.511   3.497   3.505   3.492
c** Population (MM people) ****************************************
"New England       "  13.31   0.399   0.383   0.381   0.612   0.528
"Middle Atlantic   "  38.15   0.195   0.205   0.195   0.496   0.438
"South Atlantic    "  32.83   1.240   0.971   1.051   0.442   0.624
"Florida           "  14.17   1.463   1.337   1.270   1.301   1.169
"East South Central"  16.07   1.040   0.796   0.889   0.203   0.405
"East North Central"  43.46   0.440   0.327   0.390   0.161   0.271
"West South Central"  28.83   1.166   1.098   1.043   1.066   0.984
"West North Central"  18.35   0.788   0.612   0.688   0.250   0.394
"Mountain South    "   5.90   2.436   1.707   1.860   0.406   1.105
"Mountain North    "   9.74   2.587   1.635   1.546   0.083   0.699
"California        "  32.59   0.583   1.154   0.722   2.992   1.843
"Pacific Northwest "   8.57   1.545   1.297   1.304   0.966   0.982
"Canada-East       "  21.98   1.016   0.917   0.805   0.797   0.803
"Western Canada    "   3.76   1.042   0.893   0.808   0.822   0.789
"BC-Demand         "   3.77   1.939   1.902   1.819   1.817   1.768
"Mexico-Demand     "   2.64   1.184   1.254   1.180   1.175   1.223
c** Electricity Demand (BKWh) *************************************
"TOTAL             " 3558.50  4.430   2.157   1.626   1.640   1.620
```

Table D-46

INPUT File: PROC.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the Industrial process heat demand for gas by GSAM demand region.

```
**********************************************************************
Industrial Process Heat Demand for gas(Bcf)
Data was gathered from the 1998 Industrial Trends Analysis by GRI
The data for process heat was taken directly from the book.

For GSAM regions that are subunits of Census Division, Demand was
weighted in 1995 by industrial GRP in 1995. Thus, since Pacific
Northwest made up 22 percent of the industrial GRP in the Pacific it
was given 22 percent of the feedstock demand for natural gas.
**********************************************************************
                     base    growth growth  growth  growth  growth
                     eff     rate % rate %  rate %  rate %  rate %
                     1995    1995-  2000-   2005-   2010-   2015-
                             2000   2005    2010    2015    2020
"New England       "   11.6  -1.725  1.755   1.614   1.493   1.493
"Middle Atlantic   "  192.8   4.247  1.273   0.829   0.867   0.867
"South Atlantic    "  196.2   0.772  1.959   1.399   1.428   1.428
"Florida           "   50.9   0.772  1.959   1.399   1.428   1.428
"East South Central"  148.3   2.130  3.096   1.943   2.026   2.026
"East North Central"  647.3   0.030  0.591   0.346  -0.057  -0.057
"West South Central"  986.4   1.077  1.359   1.040   0.877   0.877
"West North Central"  154.1   0.865  0.595   0.692   0.337   0.337
"Mountain South    "   43.5   1.268  1.047   0.855   0.954   0.954
"Mountain North    "   75.7   1.268  1.047   0.855   0.954   0.954
"California        "  149.5   0.599  0.389   1.036   1.160   1.160
"Pacific Northwest "   42.3   0.599  0.389   1.036   1.160   1.160
"Canada-East       "    0.0   0.000  0.000   0.000   0.000   0.000
"Western Canada    "    0.0   0.000  0.000   0.000   0.000   0.000
"BC-Demand         "    0.0   0.000  0.000   0.000   0.000   0.000
"Mexico-Demand     "    0.0   0.000  0.000   0.000   0.000   0.000
```

Table D-47

INPUT File: PROPANE.SPC (Location: \GSAM\DEMDINTG)
This file contains propane/air capacities and costs by GSAM demand region.

```
c* Propane/air Capacities and Costs
c*                  >>>>>>>>>>>>>EXISTING<<<<<<<<<<<<<<<<<<<<<  >>>>>>>>>>>>>NEW<<<<<<<<<<<<<<<<<<<<<<<<<
c*                                      Annual  Var.   Fixed                           Annual  Var.   Fixed      LOW
c*                  Delivery Stor.   1st  Demand  O&M      O&M  Deliv. Stor.   1st   Demand  O&M     O&M      BND
%
c*                  Capacity Cap.   year   Charge Charge Charge Cap.   Cap.    year  Charge  Charge Charge   OLD,
Peak
c* Demand Region     MMcfd   MMcf  avail. M$/MMcf $/Mcf   $/Mcf MMcfd   MMcf  Avail. M$/MMcf $/Mcf   $/Mcf
"New England       "   552    1327   1990    2.44   5.35   0.37   250    1000    2000    3.26   4.87   0.37  10.0
"Middle Atlantic   "   399     906   1990    2.44   5.35   0.37   250    1000    2000    3.26   4.87   0.37  10.0
"South Atlantic    "   987    4735   1990    2.44   5.35   0.37   250    1000    2000    3.26   4.87   0.37  10.0
"Florida           "    27      76   1990    2.44   5.35   0.37   250    1000    2000    3.26   4.87   0.37  10.0
"East South Central"   249    1094   1990    2.44   5.35   0.37   250    1000    2000    3.26   4.87   0.37  10.0
"East North Central"   850    3746   1990    2.44   5.35   0.37   250    1000    2000    3.26   4.87   0.37  10.0
"West South Central"     0       0   1990    0.00   0.00   0.00   250    1000    2000    3.26   4.87   0.37  10.0
"West North Central"   877    4458   1990    2.44   5.35   0.37   250    1000    2000    3.26   4.87   0.37  10.0
"Mountain South    "     1      22   1990    2.44   5.35   0.37   250    1000    2000    3.26   4.87   0.37  10.0
"Mountain North    "    21     111   1990    2.44   5.35   0.37   250    1000    2000    3.26   4.87   0.37  10.0
"California        "    42     119   1990    2.44   5.35   0.37   250    1000    2000    3.26   4.87   0.37  10.0
"Pacific Northwest "    71     112   1990    2.44   5.35   0.37   250    1000    2000    3.26   4.87   0.37  10.0
"Canada-East       "     0       0   1990    0.00   0.00   0.00   250    1000    2000    3.26   4.87   0.37  10.0
"Western Canada    "     0       0   1990    0.00   0.00   0.00   250    1000    2000    3.26   4.87   0.37  10.0
"BC-Demand         "     0       0   1990    0.00   0.00   0.00   250    1000    2000    3.26   4.87   0.37  10.0
"Mexico-Demand     "     0       0   1990    0.00   0.00   0.00   250    1000    2000    3.26   4.87   0.37  10.0
```

Table D-48

INPUT File: RES_EFF.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the residential energy efficiency by GSAM demand region.

```
c** residential energy efficiency file
c**                  base    growth growth  growth  growth  growth
c**                  eff     rate % rate %  rate %  rate %  rate %
c**                  1995    1995-  2000-   2005-   2010-   2015-
c**                  (frac)  2000   2005    2010    2015    2020
"New England       " 1.000  -0.990 -0.100  -0.100  -0.100  -0.100
"Middle Atlantic   " 1.000  -0.990 -0.100  -0.100  -0.100  -0.100
"South Atlantic    " 1.000  -0.990 -0.100  -0.100  -0.100  -0.100
"Florida           " 1.000  -0.990 -0.100  -0.100  -0.100  -0.100
"East South Central" 1.000  -0.990 -0.100  -0.100  -0.100  -0.100
"East North Central" 1.000  -0.990 -0.100  -0.100  -0.100  -0.100
"West South Central" 1.000  -0.990 -0.100  -0.100  -0.100  -0.100
"West North Central" 1.000  -0.990 -0.100  -0.100  -0.100  -0.100
"Mountain South    " 1.000  -0.990 -0.100  -0.100  -0.100  -0.100
"Mountain North    " 1.000  -0.990 -0.100  -0.100  -0.100  -0.100
"California        " 1.000  -0.990 -0.100  -0.100  -0.100  -0.100
"Pacific Northwest " 1.000  -0.990 -0.100  -0.100  -0.100  -0.100
"Canada-East       " 1.000  -0.800 -0.800  -0.800  -1.000  -1.000
"Western Canada    " 1.000  -1.000 -1.000  -1.000  -1.000  -1.000
"BC-Demand         " 1.000  -1.000 -1.000  -1.000  -1.000  -1.000
"Mexico-Demand     " 1.000  -1.000 -1.000  -1.000  -1.000  -1.000
```

Table D-49

INPUT File: RES_ELS.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the residential demand elasticities by GSAM demand region.

```
c** residential demand elasticities file
c**                  price   population energy
c**
"New England       " -0.35   0.75       1.00
"Middle Atlantic   " -0.35   0.75       1.00
"South Atlantic    " -0.35   0.75       1.00
"Florida           " -0.35   0.75       1.00
"East South Central" -0.35   0.75       1.00
"East North Central" -0.35   0.75       1.00
"West South Central" -0.35   0.75       1.00
"West North Central" -0.35   0.75       1.00
"Mountain South    " -0.35   0.75       1.00
"Mountain North    " -0.35   0.75       1.00
"California        " -0.35   0.75       1.00
"Pacific Northwest " -0.35   0.75       1.00
"Canada-East       " -0.35   0.75       1.00
"Western Canada    " -0.35   0.75       1.00
"BC-Demand         " -0.35   0.75       1.00
"Mexico-Demand     " -0.35   0.75       1.00
```

Table D-50

INPUT File: RES_LD.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the residential seasonal load factors by GSAM demand region.

```
c** residential load factors by season
c**                       season    season   season   season
c**                       1         2        3        4
"New England        "     3.64      1.94     1.72     0.58
"Middle Atlantic    "     3.21      2.09     1.78     0.55
"South Atlantic     "     3.82      1.98     1.87     0.52
"Florida            "     3.68      1.66     1.52     0.69
"East South Central"      4.30      1.97     1.89     0.50
"East North Central"      3.46      1.87     1.79     0.57
"West South Central"      4.58      1.79     1.77     0.56
"West North Central"      3.64      1.94     1.84     0.54
"Mountain South     "     3.54      1.97     1.72     0.58
"Mountain North     "     2.68      1.80     1.69     0.63
"California         "     2.90      1.62     1.48     0.72
"Pacific Northwest "      3.21      1.80     1.75     0.60
"Canada-East        "     2.73      1.91     1.72     0.60
"Western Canada     "     2.93      1.70     1.62     0.66
"BC-Demand          "     2.89      1.62     1.59     0.68
"Mexico-Demand      "     4.58      1.79     1.77     0.56
```

Table D-51

INPUT File: RES_POP.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the residential sector population by GSAM demand region.

```
c** population figures for   annual annual  annual  annual  annual
c** residential      base    growth growth  growth  growth  growth
c** sector model     popul   rate % rate %  rate %  rate %  rate %
c**                  1995    [1996, (2000,  (2005,  (2010,  (2015,
c**                  (MMP)   2000]  2005]   2010]   2015]   2020]
"New England       " 13.31    0.399  0.383   0.381   0.612   0.528
"Middle Atlantic   " 38.15    0.195  0.205   0.195   0.496   0.438
"South Atlantic    " 32.83    1.240  0.971   1.051   0.442   0.624
"Florida           " 14.17    1.463  1.337   1.270   1.301   1.169
"East South Central" 16.07    1.040  0.796   0.889   0.203   0.405
"East North Central" 43.46    0.440  0.327   0.390   0.161   0.271
"West South Central" 28.83    1.166  1.098   1.043   1.066   0.984
"West North Central" 18.35    0.788  0.612   0.688   0.250   0.394
"Mountain South    "  5.90    2.436  1.707   1.860   0.406   1.105
"Mountain North    "  9.74    2.587  1.635   1.546   0.083   0.699
"California        " 32.59    0.583  1.154   0.722   2.992   1.843
"Pacific Northwest "  8.57    1.545  1.297   1.304   0.966   0.982
"Canada-East       " 21.98    1.016  0.917   0.805   0.797   0.803
"Western Canada    "  3.76    1.042  0.893   0.808   0.822   0.789
"BC-Demand         "  3.77    1.939  1.902   1.819   1.817   1.768
"Mexico-Demand     "  2.64    1.184  1.254   1.180   1.175   1.223
```

Table D-52

INPUT File: RES_PRC.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the residential gas prices by GSAM demand region.

```
c** residential prices file  annual annual  annual  annual  annu
c**                 base    growth growth  growth  growth  grow
c**                 price   rate % rate %  rate %  rate %  rate
c**                 1995    [1996, (2000,  (2005,  (2010,  (2015,
c**                 ($/MCF) 2000]  2005]   2010]   2015]   2020]
"New England       "  8.82    1.411 -1.414  -1.545  -1.155  -1.117
"Middle Atlantic   "  7.46    2.789 -0.711  -0.912  -0.772  -0.776
"South Atlantic    "  6.90    1.600 -1.489  -1.517  -1.377  -1.337
"Florida           "  9.69   -5.071 -1.489  -1.517  -1.377  -1.337
"East South Central"  5.76    3.284 -1.087  -1.117  -1.010  -1.028
"East North Central"  5.38    2.307 -0.503  -0.725  -0.716  -0.705
"West South Central"  5.67    0.070 -0.749  -0.666  -0.380  -0.388
"West North Central"  4.92    3.805 -0.511  -0.952   0.181   0.179
"Mountain South    "  6.50    0.397 -1.556  -1.513  -1.043  -1.061
"Mountain North    "  4.08    5.766 -0.638  -1.134  -1.033  -1.090
"California        "  6.51    1.693 -1.335  -1.494  -0.527  -0.507
"Pacific Northwest "  4.12    9.883 -0.832  -1.231  -0.338  -0.344
"Canada-East       "  4.44    0.711  1.272   1.735   1.736   1.758
"Western Canada    "  3.61   -2.689  1.836   2.218   2.283   2.219
"BC-Demand         "  3.86    0.816  1.869   0.978   1.016   0.967
"Mexico-Demand     "  5.52    0.608 -0.749  -0.666  -0.380   0.000
```

Table D-53

INPUT File: SOX.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the SOx allowance costs by fuel, GSAM demand region, start year, and gas season.

```
***************************************************************************
SOx Allowance Costs ($/Kwh) by Fuel by Region by Start Year by Gas Season

***************************************************************************
Fuel/Region     Start Year Seas.1 Seas.2 Seas.3 Seas.4 Seas.5 Seas.6 Seas.7
***************************************************************************
"NUCLEAR"
"New England       "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Middle Atlantic   "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"South Atlantic    "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Florida           "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"East South Central"  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"East North Central"  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"West South Central"  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"West North Central"  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Mountain South    "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Mountain North    "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"California        "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Pacific Northwest "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Canada-East       "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Western Canada    "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"BC-Demand         "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Mexico-Demand     "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"COAL"
"New England       "  1993 0.0014 0.0014 0.0014 0.0014 0.0000 0.0000 0.0000
"Middle Atlantic   "  1993 0.0014 0.0014 0.0014 0.0014 0.0000 0.0000 0.0000
"South Atlantic    "  1993 0.0014 0.0014 0.0014 0.0014 0.0000 0.0000 0.0000
"Florida           "  1993 0.0014 0.0014 0.0014 0.0014 0.0000 0.0000 0.0000
"East South Central"  1993 0.0014 0.0014 0.0014 0.0014 0.0000 0.0000 0.0000
"East North Central"  1993 0.0014 0.0014 0.0014 0.0014 0.0000 0.0000 0.0000
"West South Central"  1993 0.0014 0.0014 0.0014 0.0014 0.0000 0.0000 0.0000
"West North Central"  1993 0.0014 0.0014 0.0014 0.0014 0.0000 0.0000 0.0000
"Mountain South    "  1993 0.0014 0.0014 0.0014 0.0014 0.0000 0.0000 0.0000
"Mountain North    "  1993 0.0014 0.0014 0.0014 0.0014 0.0000 0.0000 0.0000
"California        "  1993 0.0014 0.0014 0.0014 0.0014 0.0000 0.0000 0.0000
"Pacific Northwest "  1993 0.0014 0.0014 0.0014 0.0014 0.0000 0.0000 0.0000
"Canada-East       "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Western Canada    "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"BC-Demand         "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
"Mexico-Demand     "  2030 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
.

.

.
```

(continues with other fuels)

Table D-54

INPUT File: STORVALS.SPC (Location: \GSAM\DEMDINTG)
This file specifies the decline percentage for extraction rate from storage reservoirs and the percentage of storage capacity to be used by the model.

```
3   ! decline percentage for extraction rate from storage reservoirs
100 ! percentage of storage capacity to be used by the model
```

Table D-55

INPUT File: SUP_LD.SPC (Location: \GSAM\DEMDINTG)
This file contains data for the supply load factors by season and GSAM supply region.

```
c** supply load factors by season
c**                       season    season   season   season
c**                       1         2        3        4
"Pacific Offshore    "   1.00     1.00     1.00    1.00
"Pacific Onshore     "   0.92     0.95     1.00    1.01
"San Juan            "   0.92     0.95     1.00    1.01
"Rockies Foreland    "   0.92     0.95     1.00    1.01
"Williston           "   0.92     0.95     1.00    1.01
"Permian             "   0.92     0.95     1.00    1.01
"Mid-Continent       "   0.92     0.95     1.00    1.01
"Arkla-East Texas    "   0.92     0.95     1.00    1.01
"Texas Gulf Coast    "   0.92     0.95     1.00    1.01
"Gulf of Mexico-West "   0.93     0.93     1.00    1.01
"Gulf of Mexico-Cntr "   0.93     0.93     1.00    1.01
"Norphlet            "   1.00     1.00     1.00    1.00
"Gulf of Mexico-East "   0.92     0.95     1.00    1.01
"So-Louisiana        "   0.92     0.95     1.00    1.01
"MAFLA Onshore       "   1.00     1.00     1.00    1.00
"Mid-West            "   0.92     0.95     1.00    1.01
"Appalachia          "   0.92     0.95     1.00    1.01
"North Alaska        "   0.92     0.95     1.00    1.01
"MacKenzie Delta     "   0.92     0.95     1.00    1.01
"Alberta             "   1.00     1.00     1.00    1.00
"British Columbia    "   1.00     1.00     1.00    1.00
"Atlantic Offshore   "   0.92     0.95     1.00    1.01
"Mexico-Supply       "   0.92     0.95     1.00    1.01
```

Table D-56

INPUT File: SUPPLY.SPC (Location: \GSAM\DEMDINTG)
This file contains the supply prices and quantities by GSAM supply region. It provides the module with a supply curve. The years run from 1993-2020. The file has been abridged to fit the width of the page.

```
Pacific Offshore      4  1.92   11.0  1.84   10.0  1.75    8.0  1.71    9.0  1.91   49.4
Pacific Onshore       4  1.92  101.0  1.84   92.0  1.75   92.0  1.70   76.0  1.91  200.2
San Juan              4  1.74  861.0  1.66  879.0  1.57  953.0  1.52  975.0  1.71 1256.1
Rockies Foreland      4  1.73 1067.0  1.61 1221.0  1.49 1261.0  1.52 1311.0  1.71 1737.0
Williston             4  1.65   59.0  1.54   62.0  1.42   58.0  1.45   62.0  1.63   74.0
Permian               4  1.81  986.0  1.72 1017.0  1.63  902.0  1.59  917.0  1.77 1610.3
Mid-Continent         4  1.74 2840.0  1.66 2860.0  1.58 2697.0  1.53 2774.0  1.72 2525.5
Arkla-East Texas      4  1.85 1165.0  1.76 1164.0  1.67 1184.0  1.62 1240.0  1.81 1514.7
Texas Gulf Coast      4  1.78 1855.0  1.70 1927.0  1.61 2024.0  1.56 2155.0  1.75 2328.4
Gulf of Mexico-West   4  1.78 1206.0  1.69 1206.0  1.61 1146.0  1.56 1172.0  1.75 1341.2
Gulf of Mexico-Cntr   4  1.82 2841.0  1.74 2934.0  1.66 2847.0  1.61 3167.0  1.80 3566.8
Norphlet              4  1.84  285.6  1.76  253.7  1.67  241.4  1.63  167.2  1.83  285.6
Gulf of Mexico-East   4   .00     .0   .00     .0   .00     .0   .00     .0  3.12     .0
So-Louisiana          4  1.85  895.0  1.76  883.0  1.68  838.0  1.64  879.0  1.83  963.2
MAFLA Onshore         4  1.80  352.0  1.71  435.0  1.63  417.0  1.59  426.0  1.79  482.9
Mid-West              4  1.87  105.0  1.79  109.0  1.70  124.0  1.65  170.0  1.85   78.8
Appalachia            4  1.96  466.0  1.88  500.0  1.80  455.0  1.76  471.0  1.99  574.8
Alberta               4  1.23 4162.0  1.29 4505.0  1.44 4663.0  1.44 4900.0  1.69 4856.1
British Columbia      4  1.59  580.0  1.49  616.0  1.40  681.0  1.50  685.0  1.75  637.5
North Alaska          4   .00     .0   .00     .0   .00     .0   .00     .0   .10     .0
MacKenzie Delta       4   .00     .0   .00     .0   .00     .0   .00     .0   .96     .0
Atlantic Offshore     4   .00     .0   .00     .0   .00     .0   .00     .0   .96     .0
Mexico-Supply         4  1.80   70.0  1.72   72.0  1.64   75.0  1.53   77.0  1.72   70.0
```

Description of File: SUPPLY.SPC

Data Element	Description	Format
1	Supply region name	A20
2	Gas price track number	I3
3	Supply price ($/MCF)	F6.2
4	Supply quantity (BCF/yr.)	F7.1

(The rest of the data elements contain pairs of prices and quantities by year as in data elements 3 and 4. Prices have the same format as 3 and quantities the same format as 4. The years run from 1993-2020.)

Table D-57

INPUT File: WEATHER.SPC (Location: \GSAM\DEMDINTG)
This file contains seasonal weather-related adjustment parameters for the gas demand by GSAM demand region and year (1993-2020).

```
*********************************************************************************************
Weather File
demand = (forecast_year/base_year)^elasticity
Heating Increases Gas Demand in Res,Com,Ind Sectors Only
Cooling Increases Electricity Demand Only
*********************************************************************************************
Heating ("Winter")
                      Res   Com   Ind   Elec   Base
                      Elas  Elas  Elas  Elas   Year   1993   1994   1995   1996   1997   1998
"New England       " -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Middle Atlantic   " -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"South Atlantic    " -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Florida           " -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"East South Central" -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"East North Central" -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"West South Central" -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"West North Central" -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Mountain South    " -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Mountain North    " -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"California        " -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Pacific Northwest " -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Canada-East       " -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Western Canada    " -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"BC-Demand         " -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Mexico-Demand     " -1.00 -1.00 -1.00 0.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
*********************************************************************************************
Cooling ("Summer")
                      Res   Com   Ind   Elec   Base
                      Elas  Elas  Elas  Elas   Year   1993   1994   1995   1996   1997   1998
"New England       "  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Middle Atlantic   "  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"South Atlantic    "  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Florida           "  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"East South Central"  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"East North Central"  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"West South Central"  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"West North Central"  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Mountain South    "  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Mountain North    "  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"California        "  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Pacific Northwest "  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Canada-East       "  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Western Canada    "  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"BC-Demand         "  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
"Mexico-Demand     "  0.00  0.00  0.00 1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
```

Table D-58

OUTPUT File: GASPRC.NEW (Location: \GSAM\DEMDINTG)

This file is both an output from the Integrating Module and an Input to the E&P Module. It contains the equilibrium gas prices by year and GSAM supply region. The file has been abridged to fit the width of the page. The actual file has years from 1998 to 2020. 1 of 5 price tracks is shown.

```
Pacific Offshore      0  1   1.920   1.835   1.749   1.706   1.908   2.276   2.285   2.197   2.263   2.339
Pacific Onshore       0  1   1.920   1.835   1.749   1.696   1.908   2.269   2.279   2.194   2.259   2.336
San Juan              0  1   1.740   1.655   1.569   1.520   1.711   2.057   2.066   1.987   2.048   2.120
Rockies Foreland      0  1   1.730   1.610   1.489   1.519   1.711   2.071   2.080   2.003   2.060   2.127
Williston             0  1   1.650   1.535   1.419   1.450   1.632   1.990   1.999   1.923   1.979   2.045
Permian               0  1   1.810   1.720   1.629   1.588   1.770   2.280   2.252   2.100   2.163   2.234
Mid-Continent         0  1   1.740   1.660   1.579   1.529   1.721   2.232   2.205   2.078   2.120   2.189
Arkla-East Texas      0  1   1.850   1.760   1.669   1.617   1.809   2.315   2.287   2.135   2.198   2.268
Texas Gulf Coast      0  1   1.780   1.695   1.609   1.558   1.750   2.254   2.226   2.074   2.137   2.208
Gulf of Mexico-West   0  1   1.780   1.695   1.609   1.558   1.750   2.248   2.220   2.070   2.132   2.202
Gulf of Mexico-Cntr   0  1   1.820   1.740   1.659   1.608   1.800   2.353   2.324   2.167   2.231   2.302
Norphlet              0  1   1.840   1.755   1.669   1.627   1.829   0.529   2.358   2.195   2.260   2.332
Gulf of Mexico-East   0  1   1.820   1.740   1.659   1.608   1.800   2.622   2.599   2.430   2.499   2.563
So-Louisiana          0  1   1.850   1.765   1.679   1.637   1.829   2.387   2.357   2.199   2.263   2.334
MAFLA Onshore         0  1   1.800   1.715   1.629   1.588   1.790   2.348   2.317   2.154   2.219   2.291
Mid-West              0  1   1.870   1.785   1.699   1.648   1.849   2.426   2.395   2.231   2.297   2.372
Appalachia            0  1   1.960   1.880   1.799   1.755   1.987   2.560   2.528   2.358   2.425   2.469
Alberta               0  1   1.225   1.285   1.444   1.440   1.691   2.069   2.078   2.059   2.101   2.172
British Columbia      0  1   1.585   1.491   1.397   1.499   1.750   2.131   2.140   2.099   2.140   2.211
North Alaska          0  1   0.100   0.100   0.100   0.100   0.100   0.000   0.000   2.263   1.778   1.841
MacKenzie Delta       0  1   0.954   0.912   0.869   0.958   0.962   0.000   0.000   2.341   2.384   2.080
Atlantic Offshore     0  1   0.990   0.990   0.989   0.958   0.962   0.000   0.000   0.000   0.000   0.000
Mexico-Supply         0  1   1.800   1.720   1.639   1.529   1.721   2.228   2.200   2.045   2.109   2.180
.
.
.
```

(continues with other price tracks)

Description of File: GASPRC.NEW

Data Element	Description	Format
1	Supply region name	A20
2	Temporary index, not currently used	I3
3	Gas price track number	I3
Input supply price for time periods	($/Mcf) 1X, F7.3	

Explanation of GASPRC.NEW

This file is generated by the D&I Module and is an input to the E&P Module. It is passed to the "supply side" where the supply curve file SUPPLY.SPC is created and passed back to the "demand side". This will not happen on the first pass of an integrated run, as GASPRC.STR is substituted for GASPRC.NEW so that the E&P Module can begin.

Intended Uses of GASPRC.NEW

For the D&I Module, GASPRC.NEW is an intermediate file. The final pass of a full integrated run does produce a final gas price file, which shows the run's resulting gas prices. Note that the price tracks represent points on the supply curve, and that the second and fifth tracks show the equilibrium prices.

Table D-59

OUTPUT File: GSAMSLN.FLE (Location: \GSAM\DEMDINTG)
This output file contains supply and demand estimates and a detailed electrical power sector report. Each section has been shortened to allow the file to fit onto 3 pages in the Appendix. There are many more regions, etc. accounted for in the file than what is indicated below.

```
Supply Summary                           1993    1995    2000    2005    2010    2015
 Supply Model
 Region: Pacific Offshore                   36.9    36.9    36.9    36.9    36.9     0.0
 Region: Pacific Onshore                   271.9   272.9   294.2   262.5   211.0   123.3
 Region: San Juan                         1298.1  1256.1  1438.5  2856.8  3468.7  1548.1
 Region: Rockies Foreland                 1216.5  1801.5  3463.3  4320.8  3854.3  3737.2
 Region: Williston                          78.2   119.0   298.5   910.5  2260.0  1962.8
 Region: Permian                          1755.4  1839.7  1617.3  1449.1   963.1   493.9
 Region: Mid-Continent                    3635.5  3402.6  3011.7  2281.0  1802.0  1084.6
 Region: Arkla-East Texas                 1236.9  1442.0  1298.9   865.4   678.3   397.9
 Region: Mexico-Supply                      70.0    75.0    90.0   116.0   150.0   194.0
 Total U.S. Supply Model                 18228.0 20377.1 23021.4 24265.2 23376.2 16869.5
 Total Supply Model                      23391.7 25383.7 29204.7 31291.2 30209.3 24813.6

 Supply Projects:                         1993    1995    2000    2005    2010    2015
 Project: Sable Island                       0.0     0.0     0.0     0.0   200.0   200.0
 Project: LNG(DG) - Current                  0.0     0.0     0.0     0.0    32.9    32.9
 Project: LNG(DG) - $3.00 Thrs               0.0     0.0     0.0     0.0    20.0    20.0
 Total Supply Projects:                      0.0     0.0     0.0     0.0   331.7   331.7

 Peak Supplies:                             48.6    48.6    51.6    51.6    51.6     0.0

 Total Supplies:                         23440.3 25432.3 29256.3 31342.8 30592.6 25145.3
 --------------------------

 Demand by Region and Sector

 Demand Region: New England               1993    1995    2000    2005    2010    2015
 Sector: Residential                       176.5   202.0   217.8   214.9   210.1   200.0
 Sector: Commercial                        136.0   163.6   174.7   175.9   172.3   164.3
 Sector: Industrial                        208.4   213.0   225.0   243.4   228.4   204.1
 Sector: Elec-Gen                           27.2    36.3   266.8   267.5    89.8     0.0
 Total:                                    548.0   615.0   884.3   901.7   700.6   568.4

 Demand Region: Middle Atlantic           1993    1995    2000    2005    2010    2015
 Sector: Residential                       847.2   941.7   916.4   886.5   843.5   787.5
 Sector: Commercial                        491.4   570.0   607.3   598.5   570.4   536.2
 Sector: Industrial                        790.6   859.6   823.1   881.8   780.3   589.1
 Sector: Elec-Gen                          272.7   365.4   439.9   442.5   196.7     0.0
 Total:                                   2401.9  2736.7  2786.7  2809.3  2390.8  1912.7


 Net Demand:                             23386.8 25358.8 29170.7 31151.2 30526.0 25210.0

 Total for United States:                 1993    1995    2000    2005    2010    2015
 Sector: Residential                      4944.7  5215.8  5704.7  5779.4  5697.8  5500.0
 Sector: Commercial                       2865.6  3304.1  3633.4  3703.6  3623.4  3489.5
 Sector: Industrial                       8366.6  8616.3  9311.1 10142.1 10336.2  9494.4
 Sector: Elec-Gen                         2469.1  3056.0  4779.4  5351.2  4763.3  1325.7
 Total:                                  18645.9 20192.2 23428.6 24976.3 24420.8 19809.7

 Transport Fuel Use:                       716.2   813.8   962.1  1034.2   972.4   701.6
 Storage Fuel Use:                         128.6   128.6   128.6   128.6   128.6   128.6
 Lease & Plant:                           1184.8  1324.5  1496.4  1577.2  1519.5  1096.5

 Net Demand:                             20675.6 22459.2 26015.7 27716.4 27041.3 21736.4

 Total Imports:                           2501.2  2155.6  3079.8  3642.8  3731.7  4802.2

 Imports from Canada:                     2452.6  2107.0  3028.2  3591.1  3581.3  4703.4

 ----------------------------

 Supply Price Summary                     1993    1995    2000    2005    2010    2015
 Supply Model
 Region: Pacific Offshore                   1.48    1.66    1.24    1.09    2.39    0.00
 Region: Pacific Onshore                    1.48    1.66    1.24    1.09    2.39    5.04
 Region: San Juan                           1.32    1.48    1.09    0.95    1.99    4.65
 Region: Rockies Foreland                   1.34    1.01    0.74    0.66    1.55    4.46
 Region: Williston                          1.27    0.94    0.68    0.60    1.47    4.34
 Region: Permian                            1.47    1.59    1.22    1.29    2.44    5.13
 Region: Mid-Continent                      1.40    1.53    1.15    1.13    2.25    4.95
 Region: Arkla-East Texas                   1.50    1.62    1.25    1.33    2.48    5.17
 Region: Atlantic Offshore                  0.00    0.00    0.00    0.00    0.00    0.00
 Region: Mexico-Supply                      1.44    1.56    1.19    1.27    2.42    5.11
 Average Supply Model                       1.34    1.52    1.09    1.00    2.06    4.68

 Supply Projects:                         1993    1995    2000    2005    2010    2015
 Project: Sable Island                      0.00    0.00    0.00    0.00    1.08    5.43
 Project: LNG(DG) - Current                 0.00    0.00    0.00    0.00    2.99    5.63
```

```
Project: LNG(DG) - $3.00 Thrs             0.00    0.00    0.00    0.00    2.99    5.63
Average Supply Projects:                  0.00    0.00    0.00    0.00    1.81    5.48

Peak Supplies:                            6.46    6.03    6.87    6.73    6.59    0.00

---------------------------------

Regional Wholesale Hub and End-Use Prices by Region and Sector

Demand Region: New England              1993    1995    2000    2005    2010    2015
Sector: Residential                       8.07    8.50    8.50    8.38    9.22   12.26
Sector: Commercial                        6.21    6.61    6.58    6.46    7.35   10.29
Sector: Industrial                        2.31    2.58    2.43    2.36    3.46    5.97
Sector: Elec-Gen                          2.75    2.72    2.65    2.59    3.20    0.00
Wholesale Price:                          2.04    2.39    2.21    2.12    3.10    6.02

Industrial Petroleum Product Prices:
   Distillate:                            4.56    4.71    4.86    5.03    5.20    5.38
   Low Sulfur Resid:                      2.32    2.47    2.62    2.79    2.96    3.14
   High Sulfur Resid:                     2.03    2.18    2.33    2.50    2.67    2.85

Electrical Power Gen. Petroleum Product Prices:
   Distillate:                            3.70    3.85    4.00    4.17    4.34    4.52
   Low Sulfur Resid:                      2.34    2.49    2.64    2.81    2.98    3.16
   High Sulfur Resid:                     2.05    2.20    2.35    2.52    2.69    2.87

Demand Region: Middle Atlantic          1993    1995    2000    2005    2010    2015
Sector: Residential                       6.85    7.08    7.05    6.78    7.95   10.93
Sector: Commercial                        5.31    5.53    5.47    5.22    6.40    9.30
Sector: Industrial                        2.83    2.98    2.81    2.66    3.94    6.63
Sector: Elec-Gen                          2.19    2.37    2.28    2.19    3.07    0.00
Wholesale Price:                          1.90    2.09    1.99    1.78    3.00    5.94

Industrial Petroleum Product Prices:
   Distillate:                            4.27    4.42    4.57    4.74    4.91    5.09
   Low Sulfur Resid:                      2.57    2.72    2.87    3.04    3.21    3.39
   High Sulfur Resid:                     2.43    2.58    2.73    2.90    3.07    3.25

Electrical Power Gen. Petroleum Product Prices:
   Distillate:                            3.65    3.80    3.95    4.12    4.29    4.47
   Low Sulfur Resid:                      2.39    2.54    2.69    2.86    3.03    3.21
   High Sulfur Resid:                     2.26    2.41    2.56    2.73    2.90    3.08

Demand Region: South Atlantic           1993    1995    2000    2005    2010    2015
Sector: Residential                       6.35    6.59    6.62    6.34    7.51   10.53
Sector: Commercial                        4.90    5.12    5.10    4.86    6.04    8.93
Sector: Industrial                        2.09    2.25    2.11    1.99    3.16    5.75
Sector: Elec-Gen                          2.08    2.10    2.09    1.89    3.09    0.00
Wholesale Price:                          1.87    2.05    2.02    1.77    2.94    5.79

Industrial Petroleum Product Prices:
   Distillate:                            4.17    4.32    4.47    4.64    4.81    4.99
   Low Sulfur Resid:                      2.54    2.69    2.84    3.01    3.18    3.36
   High Sulfur Resid:                     2.13    2.28    2.43    2.60    2.77    2.95

Electrical Power Gen. Petroleum Product Prices:
   Distillate:                            3.71    3.86    4.01    4.18    4.35    4.53
   Low Sulfur Resid:                      2.36    2.51    2.66    2.83    3.00    3.18
   High Sulfur Resid:                     1.97    2.12    2.27    2.44    2.61    2.79


Region: New England         Electric Generation Model Results

Generation Type: Nuclear                1993    1995    2000    2005    2010    2015
Existing Capacity: (gigawatts)            6.67    6.38    6.38    6.38    4.30    2.27
New Capacity: (gigawatts)                 0.00    0.00    0.00    0.00    0.00    0.00
Total Capacity:                           6.67    6.38    6.38    6.38    4.30    2.27
Capacity Utilization
Period: Peak Days     Gas:                 0.0     0.0     0.0     0.0     0.0     0.0
                      Other:              63.9    63.9    63.9    63.9    63.9    64.0
Period: Peak Season   Gas:                 0.0     0.0     0.0     0.0     0.0     0.0
                      Other:              63.9    63.9    63.9    63.9    63.9    63.9
Period: Shoulder      Gas:                 0.0     0.0     0.0     0.0     0.0     0.0
                      Other:              63.9    63.9    63.9    63.9    63.9    63.9
Period: Base          Gas:                 0.0     0.0     0.0     0.0     0.0     0.0
                      Other:              63.9    63.9    63.9    63.9    63.9    63.9

Generation Type: Coal                   1993    1995    2000    2005    2010    2015
Existing Capacity: (gigawatts)            2.62    2.64    2.64    2.59    2.59    2.59
New Capacity: (gigawatts)                 0.00    0.00    0.10    1.05    2.27    6.66
Total Capacity:                           2.62    2.64    2.74    3.64    4.86    9.25
Capacity Utilization
Period: Peak Days     Gas:                 0.0     0.0     0.0     0.0     0.0     0.0
                      Other:              70.2    70.2    70.2    70.5    70.6    70.8
Period: Peak Season   Gas:                 0.0     0.0     0.0     0.0     0.0     0.0
                      Other:              70.2    70.2    70.2    70.4    70.6    70.8
Period: Shoulder      Gas:                 0.0     0.0     0.0     0.0     0.0     0.0
                      Other:              70.2    70.2    70.2    70.4    70.6    70.8
Period: Base          Gas:                 0.0     0.0     0.0     0.0     0.0     0.0
                      Other:              70.2    70.2    70.2    70.4    70.6    70.8
```

```
Total Generation:                    1993    1995    2000    2005    2010    2015
Existing Capacity: (gigawatts)        24.24   23.28   21.83   21.32   19.20   16.88
New Capacity: (gigawatts)              3.62    4.21    6.32    7.89    9.11   13.50
Total Capacity:                       27.86   27.49   28.15   29.21   28.31   30.38

Non-Gas Generation (billion kwhs)      71.7    72.5    62.5    68.1    79.7    94.3

Gas Generation (billion kwhs)           2.5     3.7    26.3    26.3    10.6     0.0
Overall Capacity Factor(%)             30.4    31.7    36.0    36.9    36.4    35.4

-------------------------------------------------
```

Table D-60

OUTPUT File: GSAMSLN.RPT (Location: \GSAM\DEMDINTG)
This output file has information on transportation flows and capacities. Each section of the file has been considerably shortened in order to fit to 3 pages in the Appendix. There are many more pipelines modeled than are indicated by the file below.

```
 Transportation Capacity Summary Report (MMcf/day)
 Origin              Destination            1993     1995     2000     2005     2010     2015

 East South Central  Appalachia           2361.00  2361.00  2361.00  2361.00  2361.00  2361.00
 North Alaska        Alberta                 0.00     0.00     0.00     0.00     0.00    82.05
 MacKenzie Delta     Alberta                 0.00     0.00     0.00     0.00     0.00     0.00
 British Columbia    Alberta               220.00   220.00   220.00   220.00   220.00   220.00
 Gulf of Mexico-East East South Central   7004.00  7004.00  7004.00  7004.00  7004.00  7004.00
 Gulf of Mexico-East Gulf of Mexico-West  2600.00  2600.00  2600.00  2600.00  2600.00  2600.00
 Alliance-Supply     East North Central      0.00     0.00     0.00     0.00     0.00     0.00

 --------------------------------------------

 Trans. Flow Report for L-Period:  1 Days: 151 (mmcf/day)
 Origin              Destination            1993     1995     2000     2005     2010     2015

 East South Central  Appalachia           2361.00  2361.00  2361.00  2361.00    15.37     0.00
 North Alaska        Alberta                 0.00     0.00     0.00     0.00     0.00    82.05
 MacKenzie Delta     Alberta                 0.00     0.00     0.00     0.00     0.00     0.00
 British Columbia    Alliance-Supply         0.00     0.00     0.00     0.00     0.00     0.00
 Alberta             Alliance-Supply         0.00     0.00     0.00     0.00     0.00     0.00
 British Columbia    Alberta              -220.00     0.00  -220.00  -220.00   -77.00  -220.00
 Gulf of Mexico-East East South Central   7004.00  7004.00  7004.00  7004.00  7004.00  5798.25
 Gulf of Mexico-East Gulf of Mexico-West -2600.00 -2600.00 -2600.00 -2600.00 -2600.00 -2600.00
 Alliance-Supply     East North Central      0.00     0.00     0.00     0.00     0.00     0.00

--------------------------------------------

 Trans. Flow Report - Annual (Bcf)
 Origin              Destination            1993     1995     2000     2005     2010     2015

 East South Central  Appalachia            861.77   861.77   861.77   861.77   461.89     0.00
 North Alaska        Alberta                 0.00     0.00     0.00     0.00     0.00    29.95
 MacKenzie Delta     Alberta                 0.00     0.00     0.00     0.00     0.00     0.00
 British Columbia    Alliance-Supply         0.00     0.00     0.00     0.00     0.00     0.00
 Alberta             Alliance-Supply         0.00     0.00     0.00     0.00     0.00     0.00
 British Columbia    Alberta               -33.22     0.00   -33.22   -33.22   -11.63   -33.22
 Gulf of Mexico-East East South Central   2556.46  2556.46  2556.46  2556.46  2273.88  1386.22
 Gulf of Mexico-East Gulf of Mexico-West  -949.00  -949.00  -949.00  -949.00  -949.00  -949.00
 Alliance-Supply     East North Central      0.00     0.00     0.00     0.00     0.00     0.00

 --------------------------------------------

 Trans. Flow Report - Maximum Utilization (%)
 Origin              Destination            1993     1995     2000     2005     2010     2015

 East South Central  Appalachia            100.00   100.00   100.00   100.00    90.96     0.00
 North Alaska        Alberta                 0.00     0.00     0.00     0.00     0.00   100.00
 MacKenzie Delta     Alberta                 0.00     0.00     0.00     0.00     0.00     0.00
 Pacific Northwest   California            100.00    87.41   100.00   100.00   100.00   100.00
 British Columbia    Alberta               100.00     0.00   100.00   100.00    35.00   100.00
 Gulf of Mexico-East East South Central    100.00   100.00   100.00   100.00   100.00    82.78
 Gulf of Mexico-East Gulf of Mexico-West   100.00   100.00   100.00   100.00   100.00   100.00
 Alliance-Supply     East North Central      0.00     0.00     0.00     0.00     0.00     0.00

 --------------------------------------------
 Node Flow Report for : New England

 Load Period:  1 Number of Days:151 (mmcf/day)
                                            1993     1995     2000     2005     2010     2015
 Transport to Node from Specified Location
 Canada-East                                63.00    63.00   744.28   790.70   790.70   790.70
 Distrigas                                   0.00     0.00     0.00     0.00   284.38   284.38
 Middle Atlantic                          1984.42  2210.00  2210.00  2210.00   730.71   527.45
 Sable Island                                0.00     0.00     0.00     0.00   547.95   547.95
 Total Flows In                           2047.42  2273.00  2954.27  3000.70  2353.74  2150.47

 Fuel Use on Transport to Node
 Canada-East                                 0.97     0.97    11.46    12.18    12.18    12.18
 Distrigas                                   0.00     0.00     0.00     0.00     0.00     0.00
 Middle Atlantic                            99.02   110.28   110.28   110.28    36.46    26.32
 Sable Island                                0.00     0.00     0.00     0.00    16.44    16.44
 Fuel Use In Transportation                 99.99   111.25   121.74   122.46    65.08    54.93

 Net Transport to Node                    1947.43  2161.75  2832.53  2878.24  2288.66  2095.54

 Storage Extraction
 Total Storage Extraction:                   0.00     0.00     0.00     0.00     0.00     0.00

 Peaking Supply Source
 Propane                                     4.37     4.37     4.37     4.37     4.37     0.00
 LNG                                        26.82    26.82    33.44    33.44    33.44     0.00
 Total Peaking Supply:                      31.19    31.19    37.81    37.81    37.81     0.00

 Total Interuption:                        677.57   835.38   648.26   665.45  1089.06   895.92
 Total Supply:                            2656.19  3028.32  3518.60  3581.50  3415.52  2991.46

 Transport from Node to Specified Location
 Canada-East                                 0.00     0.00     0.00     0.00     0.00     0.00
 Distrigas                                   0.00     0.00     0.00     0.00     0.00     0.00
 Middle Atlantic                             0.00     0.00     0.00     0.00     0.00     0.00
 Sable Island                                0.00     0.00     0.00     0.00     0.00     0.00
 Total Flows Out                              0.00     0.00     0.00     0.00     0.00     0.00

 Storage Injection
 Total Storage Injection:                     0.00     0.00     0.00     0.00     0.00     0.00

 Customer Demand:                          2656.17  3028.30  3518.58  3581.48  3415.50  2991.44

 Lease and Plant Usage:                       0.00     0.00     0.00     0.00     0.00     0.00
 Total Demand:                             2656.17  3028.30  3518.58  3581.48  3415.50  2991.44

***************************************************
 Total Supply-Total Demand:                   0.02     0.02     0.02     0.02     0.03     0.02
***************************************************


 Load Period:  2 Number of Days:214 (MMcf/day)
                                            1993     1995     2000     2005     2010     2015
 Transport to Node from Specified Location
 Canada-East                                 63.00    63.00   744.28   790.70   790.70   367.15
 Distrigas                                    0.00     0.00     0.00     0.00   284.38   284.38
 Middle Atlantic                           1160.56  1330.62  1446.04  1449.96    39.83     0.00
 Sable Island                                 0.00     0.00     0.00     0.00   547.95   547.95
 Total Flows In                            1223.56  1393.62  2190.31  2240.66  1662.86  1199.48

 Fuel Use on Transport to Node
 Canada-East                                  0.97     0.97    11.46    12.18    12.18     5.65
 Distrigas                                    0.00     0.00     0.00     0.00     0.00     0.00
 Middle Atlantic                             57.91    66.40    72.16    72.35     1.99     0.00
 Sable Island                                 0.00     0.00     0.00     0.00    16.44    16.44
 Fuel Use In Transportation                  58.88    67.37    83.62    84.53    30.60    22.09

 Net Transport to Node                     1164.68  1326.25  2106.70  2156.13  1632.25  1177.38

 Storage Extraction
 Total Storage Extraction:                    0.00     0.00     0.00     0.00     0.00     0.00

 Peaking Supply Source
 Propane                                      0.00     0.00     0.00     0.00     0.00     0.00
 LNG                                          0.00     0.00     0.00     0.00     0.00     0.00
 Total Peaking Supply:                        0.00     0.00     0.00     0.00     0.00     0.00

 Total Interuption:                         358.68   412.30     0.00     0.00   407.14   514.06
 Total Supply:                             1523.36  1738.55  2106.70  2156.13  2039.40  1691.44

 Transport from Node to Specified Location
 Canada-East                                  0.00     0.00     0.00     0.00     0.00     0.00
 Distrigas                                    0.00     0.00     0.00     0.00     0.00     0.00
 Middle Atlantic                              0.00     0.00     0.00     0.00     0.00     0.00
 Sable Island                                 0.00     0.00     0.00     0.00     0.00     0.00
 Total Flows Out                              0.00     0.00     0.00     0.00     0.00     0.00

 Storage Injection
 Total Storage Injection:                     0.00     0.00     0.00     0.00     0.00     0.00

 Customer Demand:                          1523.37  1738.57  2106.71  2156.14  2039.41  1691.46

 Lease and Plant Usage:                       0.00     0.00     0.00     0.00     0.00     0.00
 Total Demand:                             1523.37  1738.57  2106.71  2156.14  2039.41  1691.46

***************************************************
 Total Supply-Total Demand:                  -0.02    -0.02    -0.01    -0.01    -0.01    -0.01
***************************************************


 Annual Averages (Bcf)
                                            1993     1995     2000     2005     2010     2015
 Transport to Node from Specified Location
 Canada-East                                 22.99    22.99   271.66   288.60   288.60   197.96
 Distrigas                                    0.00     0.00     0.00     0.00   103.80   103.80
 Middle Atlantic                            548.01   618.46   643.16   644.00   118.86    79.64
 Sable Island                                 0.00     0.00     0.00     0.00   200.00   200.00
 Total Flows In                             571.00   641.46   914.82   932.61   711.27   581.41

 Fuel Use on Transport to Node
 Canada-East                                  0.35     0.35     4.18     4.44     4.44     3.05
 Distrigas                                    0.00     0.00     0.00     0.00     0.00     0.00
 Middle Atlantic                             27.35    30.86    32.09    32.14     5.93     3.97
 Sable Island                                 0.00     0.00     0.00     0.00     6.00     6.00
 Fuel Use In Transportation                  27.70    31.22    36.28    36.58    16.38    13.02

 Net Transport to Node                      543.30   610.24   878.55   896.03   694.89   568.39

 Storage Extraction
 Total Storage Extraction:                    0.00     0.00     0.00     0.00     0.00     0.00

 Peaking Supply Source
 Propane                                      0.66     0.66     0.66     0.66     0.66     0.00
 LNG                                          4.05     4.05     5.05     5.05     5.05     0.00
 Total Peaking Supply:                        4.71     4.71     5.71     5.71     5.71     0.00

 Total Interuption:                         179.07   214.38    97.89   100.48   251.58   245.29
 Total Supply:                              727.08   829.33   982.14  1002.22   952.17   813.68

 Transport from Node to Specified Location
 Canada-East                                  0.00     0.00     0.00     0.00     0.00     0.00
 Distrigas                                    0.00     0.00     0.00     0.00     0.00     0.00
 Middle Atlantic                              0.00     0.00     0.00     0.00     0.00     0.00
 Sable Island                                 0.00     0.00     0.00     0.00     0.00     0.00
 Total Flows Out                              0.00     0.00     0.00     0.00     0.00     0.00

 Storage Injection
 Total Storage Injection:                     0.00     0.00     0.00     0.00     0.00     0.00

 Customer Demand:                           727.08   829.33   982.14  1002.22   952.17   813.68

 Lease and Plant Usage:                       0.00     0.00     0.00     0.00     0.00     0.00
 Total Demand:                              727.08   829.33   982.14  1002.22   952.17   813.68
```

```
****************************************************
 Total Supply-Total Demand:                   0.00     0.00     0.00     0.00     0.00     0.00
****************************************************
```

Table D-61

OUTPUT File: GSAMSLN.STA (Location: \GSAM\DEMDINTG)
This file contains a summary of storage activity. It has been considerably shortened to fit to one page in the Appendix.

```
 ===============================================

 Storage Activity Summary Report

 ===============================================


 --------------------------------------------
 Demand Region : Middle Atlantic

 Load Period:  1 Number of Days:151 (MMcf/day)
                                        1993     1995     2000     2005     2010     2015
 Storage Extraction
                         Avail. Cap.      Storage Usage
      02706719218        2.16               1.99     1.99     1.99     1.99     1.99     1.99
      02706720448       10.36              13.38    13.38    13.38    13.38    11.49     9.87
      02706720449       10.36              13.38    13.38    13.38    13.38    11.49     9.87
      02706721201       15.04              11.66    11.66    11.66    11.66    11.66    11.66
Total Storage Extraction:                1462.47  1467.30  2284.17  2238.00  2143.75  2192.64

 Storage Injection
      02706719218        1.49               0.00     0.00     0.00     0.00     0.00     0.00
      02706720448        7.12               0.00     0.00     0.00     0.00     0.00     0.00
      02706720449        7.12               0.00     0.00     0.00     0.00     0.00     0.00
      02706721201       10.27               0.00     0.00     0.00     0.00     0.00     0.00
Total Storage Injection:                    0.00     0.00     0.00     0.00     0.00     0.00

 Load Period:  2 Number of Days:214 (MMcf/day)
                                        1993     1995     2000     2005     2010     2015
 Storage Extraction
                         Avail. Cap.      Storage Usage
      02706719218        2.16               0.00     0.00     0.00     0.00     0.00     0.00
      02706720448       10.36               0.00     0.00     0.00     0.00     0.00     0.00
      02706720449       10.36               0.00     0.00     0.00     0.00     0.00     0.00
      02706721201       15.04               0.00     0.00     0.00     0.00     0.00     0.00

Total Storage Extraction:                   0.00     0.00     0.00     0.00     0.00     0.00

 Storage Injection
      02706719218        1.49               1.43     1.43     1.43     1.43     1.43     1.43
      02706720448        7.12               9.65     9.65     9.65     9.65     8.29     7.12
      02706720449        7.12               9.65     9.65     9.65     9.65     8.29     7.12
      02706721201       10.27               8.41     8.41     8.41     8.41     8.41     8.41
Total Storage Injection:               1 055.36  1058.85  1648.32  1615.00  1546.98  1582.27

 Annual Averages (Bcf) Note: Storage Injection
Values Are **Not** Less Fuel Usage
                                        1993     1995     2000     2005     2010     2015
 Storage Extraction
                         Avail. Cap.      Storage Usage
      02706719218        0.30               0.30     0.30     0.30     0.30     0.30     0.30
      02706721201        1.76               1.76     1.76     1.76     1.76     1.76     1.76
Total Storage Extraction:                 220.83   221.56   344.91   337.94   323.71   331.09

 Storage Injection
      02706720448        6.32               2.07     2.07     2.07     2.07     1.77     1.52
      02706720449        6.32               2.07     2.07     2.07     2.07     1.77     1.52

Total Storage Injection:                  225.85   226.59   352.74   345.61   331.05   338.61
```

Table D-62

Output File: GSAMSLN.STC (Location: \GSAM\DEMDINTG)
This output file contains a summary of storage costs. It has been shortened to fit to one Appendix page.

```
 ==========================================

 Storage Costs Summary Report

 ==========================================


 ------------------------------------------
 Demand Region : New England

 Annual Values ($/MCF)
                                        1993     1995     2000     2005     2010     2015

 ------------------------------------------
 Demand Region : Middle Atlantic

 Annual Values ($/MCF)
                                        1993     1995     2000     2005     2010     2015
      02706719218                       0.90     0.90     0.90     0.90     0.92     0.99
      02706720448                       0.58     0.58     0.58     0.57     0.60     0.67
      02706720449                       0.58     0.58     0.58     0.57     0.60     0.67
      02706721201                       0.48     0.49     0.48     0.48     0.51     0.57
      02706721203                       0.48     0.49     0.48     0.48     0.51     0.57


 ------------------------------------------
 Demand Region : South Atlantic

 Annual Values ($/MCF)
                                        1993     1995     2000     2005     2010     2015
      03706720138                       0.32     0.32     0.32     0.32     0.34     0.41
      03706721345                       0.48     0.49     0.48     0.48     0.51     0.57
      03706721346                       0.48     0.49     0.48     0.48     0.51     0.57
      03706721347                       0.48     0.49     0.48     0.48     0.51     0.57
      03706721349                       0.48     0.49     0.48     0.48     0.51     0.57


 Demand Region : East South Central

 Annual Values ($/MCF)
                                        1993     1995     2000     2005     2010     2015
      05704941190                       0.55     0.55     0.55     0.55     0.56     0.60
      05706401119                       0.78     0.79     0.78     0.78     0.80     0.83
      05706401120                       0.78     0.79     0.78     0.78     0.80     0.83
      05706401121                       0.78     0.79     0.78     0.78     0.80     0.83


 ------------------------------------------
 Demand Region : East North Central

 Annual Values ($/MCF)
                                        1993     1995     2000     2005     2010     2015
      06706301154                    9999.99  9999.99  9999.99  9999.99  9999.99  9999.99
      06706301173                       0.48     0.48     0.48     0.48     0.49     0.53
      06706303145                       0.67     0.67     0.67     0.67     0.68     0.71
      06706303146                       0.67     0.67     0.67     0.67     0.68     0.71

 ------------------------------------------
```

Table D-63

OUTPUT File: GSAMSLN.SEA (Location: \GSAM\DEMDINTG)
This output file contains report on seasonal output by GSAM region, sector, season, and year. The demand section is given below.

```
Seasonal Output Report
-------------------------

1998
Demand by Region, Sector, and Season (MMcfd)

                                            1998
Region: New England              1       2       3       4  Ann(Bcf)
Sector: Residential          841.6   448.5   397.7   134.1   165.9
Sector: Commercial           608.4   466.7   404.0   185.5   151.7
Sector: Industrial           226.8   706.6   767.0   835.5   231.6
Sector: Elec-Gen               0.0     0.0     0.0   697.5    64.2
Total:                      1676.8  1621.8  1568.7  1852.6   613.4

                                            1998
Region: Middle Atlantic          1       2       3       4  Ann(Bcf)
Sector: Residential         3629.1  2362.9  2012.4   621.8   785.6
Sector: Commercial          1941.7  1762.2  1501.7   634.7   532.1
Sector: Industrial          1941.4  2937.3  2867.9  2513.0   936.1
Sector: Elec-Gen               0.0     0.0     0.0  2407.8   221.5
Total:                      7512.1  7062.4  6382.0  6177.3  2475.4

                                            1998
Region: South Atlantic           1       2       3       4  Ann(Bcf)
Sector: Residential         1934.1  1002.5   946.8   263.3   377.6
Sector: Commercial          1076.0   854.4   754.2   325.7   274.3
Sector: Industrial          1040.4  1725.7  1799.2  2009.6   600.3
Sector: Elec-Gen               0.0     0.0     0.0    82.2     7.6
Total:                      4050.5  3582.6  3500.1  2680.8  1259.8

                                            1998
Region: Florida                  1       2       3       4  Ann(Bcf)
Sector: Residential           77.5    35.0    32.0    14.5    14.5
Sector: Commercial           158.0   115.4   108.9    83.9    42.5
Sector: Industrial           218.0   297.6   301.5   320.4   103.8
Sector: Elec-Gen               0.0   540.0   590.5  1156.4   209.3
Total:                       453.6   988.0  1032.9  1575.2   370.1

                                            1998
Region: East South Central       1       2       3       4  Ann(Bcf)
Sector: Residential         1024.0   469.1   450.1   119.1   187.8
Sector: Commercial           575.7   458.3   391.6   150.1   143.5
Sector: Industrial           897.9  1142.7  1125.2  1052.4   384.9
Sector: Elec-Gen               0.0     0.0     0.0   126.6    11.7
Total:                      2497.6  2070.1  1966.9  1448.2   727.9

                                            1998
Region: East North Central       1       2       3       4  Ann(Bcf)
Sector: Residential         6991.1  3778.5  3616.8  1151.7  1415.1
Sector: Commercial          3271.7  2292.8  2108.5   734.2   765.8
Sector: Industrial          5816.6  7495.0  7090.6  5625.5  2374.1
Sector: Elec-Gen               0.0     0.0     0.0    14.0     1.3
Total:                     16079.5 13566.2 12815.9  7525.4  4556.3

                                            1998
Region: West South Central       1       2       3       4  Ann(Bcf)
Sector: Residential         1988.9   777.3   768.7   243.2   344.1
Sector: Commercial          1198.4   895.0   840.4   470.1   310.2
Sector: Industrial          5562.6  7626.5  7495.5  7659.2  2586.9
Sector: Elec-Gen               0.0   442.1   677.1   605.1   157.5
Total:                      8749.9  9740.9  9781.7  8977.6  3398.7

                                            1998
Region: West North Central       1       2       3       4  Ann(Bcf)
Sector: Residential         2245.1  1196.6  1134.9   333.1   447.1
Sector: Commercial          1226.8   985.7   889.7   335.2   313.1
Sector: Industrial           945.0  1210.6  1182.3  1041.7   399.6
Sector: Elec-Gen               0.0     0.0     0.0    70.1     6.5
Total:                      4416.9  3392.9  3206.9  1780.1  1166.3

                                            1998
Region: Mountain South           1       2       3       4  Ann(Bcf)
Sector: Residential          277.1   154.2   134.6    45.4    55.7
Sector: Commercial           202.5   158.1   145.4    84.0    53.8
Sector: Industrial           175.3   189.6   179.0   174.4    65.5
Sector: Elec-Gen               0.0     0.0     0.0    67.4     6.2
Total:                       654.8   501.9   459.0   371.2   181.2
```

```
                                                1998
Region: Mountain North           1       2       3       4   Ann(Bcf)
Sector: Residential          873.4   586.6   550.7   205.3   201.9
Sector: Commercial           534.6   438.0   402.6   172.4   141.0
Sector: Industrial           747.3   864.3   811.7   717.3   286.5
Sector: Elec-Gen               0.0    65.4   248.4   616.8    85.3
Total:                      2155.3  1954.4  2013.5  1711.9   714.7

                                                1998
Region: California               1       2       3       4   Ann(Bcf)
Sector: Residential         2186.5  1221.4  1115.9   542.9   461.6
Sector: Commercial          1070.3   830.9   753.6   574.0   294.4
Sector: Industrial           848.0  1268.2  1295.5  1418.3   440.9
Sector: Elec-Gen               0.0   442.9   693.9  2546.0   337.7
Total:                      4104.8  3763.5  3858.9  5081.1  1534.6

                                                1998
Region: Pacific Northwest        1       2       3       4   Ann(Bcf)
Sector: Residential          339.6   190.5   185.2    63.5    70.9
Sector: Commercial           247.9   185.0   179.2    80.0    63.1
Sector: Industrial           303.1   400.5   398.2   385.7   135.7
Sector: Elec-Gen               0.0     0.0     0.0   100.3     9.2
Total:                       890.7   775.9   762.5   629.5   279.0

                                                1998
Region: Canada-East              1       2       3       4   Ann(Bcf)
Sector: Residential         1185.4   829.4   746.9   260.5   275.3
Sector: Commercial          1305.2  1054.3   945.5   338.4   331.9
Sector: Industrial           730.3  2523.9  2419.7  1883.2   689.6
Sector: Elec-Gen               0.0     0.0     0.0   115.9    10.7
Total:                      3220.8  4407.5  4112.1  2598.0  1307.4

                                                1998
Region: Western Canada           1       2       3       4   Ann(Bcf)
Sector: Residential          663.9   385.2   367.1   149.6   142.6
Sector: Commercial           425.7   273.7   260.7   109.4    97.4
Sector: Industrial           449.9  1332.0  1306.8  1307.1   401.3
Sector: Elec-Gen               0.0    31.6   174.2   371.9    52.9
Total:                      1539.5  2022.5  2108.8  1938.0   694.3

                                                1998
Region: BC-Demand                1       2       3       4   Ann(Bcf)
Sector: Residential          296.7   166.3   163.2    69.8    63.4
Sector: Commercial           225.8   152.6   144.0    67.4    53.7
Sector: Industrial           103.1   294.7   264.2   310.8    88.8
Sector: Elec-Gen               0.0     0.0    15.5   124.9    12.9
Total:                       625.6   613.6   586.9   573.0   218.9

                                                1998
Region: Mexico-Demand            1       2       3       4   Ann(Bcf)
Sector: Residential          144.3    56.4    55.8    17.6    25.0
Sector: Commercial             5.6     4.2     3.9     2.2     1.4
Sector: Industrial            31.3    96.3    93.6   103.5    29.6
Sector: Elec-Gen               0.0    49.5    86.2    83.4    20.0
Total:                       181.2   206.3   239.5   206.6    76.1
.
.
.
```
(continues with other indicators)

Table D-64

Output File: GSAMSLN.SUP (Location: \GSAM\DEMDINTG)
This output file contains report on supply sources by load period, GSAM region and year.

```
 Decline factor:    3.0%  Storage Usage: 100.0%
 Demand Region : New England

>>>>>>>>>>>>>>>>>>>> Load Period:  1 Number of Days: 91 (MMCF/day)<<<<<<<<<<<<<<<<<<<<
                                           1998     1999     2000     2001     2002     2003     2004     2005     2006     2007     2008     2009     2010     2011     2012     2013     2014     2015     2016     2017     2018     2019     2020

 Peaking Supply Sources:
Propane:
 Existing   Usage                        265.40   265.40    55.20    55.20    55.20    55.20    55.20    55.20    55.20    55.20    55.20    55.20    55.20    55.20    60.45   265.40   265.40   265.40   265.40   265.40    55.20    55.20    55.20
 Existing   Maximum                      552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00   552.00
 Existing   Price ($/MCF)                  5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73     5.73

 New        Usage                          0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00   200.00   200.00   200.00   200.00   200.00   200.00   200.00   200.00    94.74
 New        Maximum                      250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00   250.00
 New        Price ($/MCF)               99999.99 99999.99 99999.99 99999.99 99999.99 99999.99 99999.99 99999.99 99999.99 99999.99 99999.99 99999.99 99999.99 99999.99     5.25     5.25     5.25     5.25     5.25     5.25     5.25     5.25     5.25

 Total      Usage                        265.40   265.40    55.20    55.20    55.20    55.20    55.20    55.20    55.20    55.20    55.20    55.20    55.20    55.20   260.45   465.40   465.40   465.40   465.40   465.40   255.20   255.20   149.94
 Total      Maximum                      802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00   802.00
LNG:
 Existing   Usage                       1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00
 Existing   Maximum                     1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00  1400.00
 Existing   Price ($/MCF)                  2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24     2.24

 New        Usage                          0.00     0.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00
 New        Maximum                      400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00   400.00
 New        Price ($/MCF)               99999.99 99999.99     1.32     1.32     1.32     1.32     1.32     1.32     1.32     1.32     1.32     1.32     1.32     1.32     1.32     1.32     1.32     1.32     1.32     1.32     1.32     1.32     1.32

 Total      Usage                       1400.00  1400.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00
 Total      Maximum                     1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00  1800.00

 Total Peaking Supply Usage:            1665.40  1665.40  1855.20  1855.20  1855.20  1855.20  1855.20  1855.20  1855.20  1855.20  1855.20  1855.20  1855.20  1855.20  2060.45  2265.40  2265.40  2265.40  2265.40  2265.40  2055.20  2055.20  1949.94
 Total Storage Extraction:                 0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
 Maximum Possible:                         0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
 Avg. Price ($/MCF)                        0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
 Total Pipeline Flows In                1160.67  1182.89  1666.69  1694.24  1703.75  1712.46  1739.31  1731.74  1741.20  1758.09  1798.53  1852.18  1887.26  1873.15  1691.96  1519.95  1547.49  1593.69  1627.22  1668.65  1957.73  1994.05  2161.11
 Total Pipeline Flows Out                  0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
 Net Pipeline Flows In                  1160.67  1182.89  1666.69  1694.24  1703.75  1712.46  1739.31  1731.74  1741.20  1758.09  1798.53  1852.18  1887.26  1873.15  1691.96  1519.95  1547.49  1593.69  1627.22  1668.65  1957.73  1994.05  2161.11
 Maximum Capacity                       2705.00  3033.00  3433.00  3433.00  3433.00  3433.00  3433.00  3833.00  3833.00  3833.00  3833.00  3833.00  3833.00  3833.00  3833.00  3833.00  3833.00  4233.00  4233.00  4233.00  4233.00  4233.00  4633.00
```

•

•

(continues with other load periods and regions)

Table D-65

OUTPUT File: GASALL.PRT (Location: \GSAM\DEMDINTG)
This file contains the solution from the LP software (ASCII format). It has been considerably shortened as it is too long.

```
Problem Statistics
Matrix Integrat
Objective OBJ
RHS RHS1
Problem has  82077 rows an 1940431 structural columns

Solution Statistics
Minimization performed
Optimal solution found after     44 iterations
Objective function value is -1114527.998


Rows Section
   Number     Row    At        Value     Slack Value     Dual Value      RHS
 N      1  OBJ       BS  -1114527.998  1114527.998       .000000       .000000
 G      2  MB050     LL       .000000       .000000       .222167       .000000
 G      3  MB051     LL       .000000       .000000       .077872       .000000
 G      4  MB052     LL       .000000       .000000       .269439       .000000
 G      5  MB053     LL       .000000       .000000       .568210       .000000
 G      6  MB060     LL       .000000       .000000       .147986       .000000
 G      7  MB061     LL       .000000       .000000       .071292       .000000
 G      8  MB062     LL       .000000       .000000       .242730       .000000
 G      9  MB063     LL       .000000       .000000       .522734       .000000
 G     10  MB070     LL       .000000       .000000       .014827       .000000
 G     11  MB071     LL       .000000       .000000       .070640       .000000
 G     12  MB072     LL       .000000       .000000       .225541       .000000
 G     13  MB073     LL       .000000       .000000       .491888       .000000
 G     14  MB080     LL       .000000       .000000       .013186       .000000
 G     15  MB081     LL       .000000       .000000       .064897       .000000
 G     16  MB082     LL       .000000       .000000       .193263       .000000
 G     17  MB083     LL       .000000       .000000       .456256       .000000
 G     18  MB090     LL       .000000       .000000       .012970       .000000
 G     19  MB091     LL       .000000       .000000       .052366       .000000
 G     20  MB092     LL       .000000       .000000       .165054       .000000
 G     21  MB093     LL       .000000       .000000       .406006       .000000
   Number     Row    At        Value     Slack Value     Dual Value      RHS
 G     22  MB0A0     LL       .000000       .000000       .011279       .000000
 G     23  MB0A1     LL       .000000       .000000       .048026       .000000
 G     24  MB0A2     LL       .000000       .000000       .163506       .000000
 G     25  MB0A3     LL       .000000       .000000       .371435       .000000
 G     26  MB0B0     LL       .000000       .000000       .010848       .000000
 G     27  MB0B1     LL       .000000       .000000       .042321       .000000
 G     28  MB0B2     LL       .000000       .000000       .144202       .000000
 G     29  MB0B3     LL       .000000       .000000       .331660       .000000
 G     30  MB0C0     LL       .000000       .000000       .009570       .000000
 G     31  MB0C1     LL       .000000       .000000       .037903       .000000
 G     32  MB0C2     LL       .000000       .000000       .131036       .000000
 G     33  MB0C3     LL       .000000       .000000       .285121       .000000
 G     34  MB0D0     LL       .000000       .000000       .010210       .000000
 G     35  MB0D1     LL       .000000       .000000       .035541       .000000
 G     36  MB0D2     LL       .000000       .000000       .117189       .000000
 G     37  MB0D3     LL       .000000       .000000       .269280       .000000
 G     38  MB0E0     LL       .000000       .000000       .009670       .000000
 G     39  MB0E1     LL       .000000       .000000       .031903       .000000
 G     40  MB0E2     LL       .000000       .000000       .112116       .000000
 G     41  MB0E3     LL       .000000       .000000       .244207       .000000
 G     42  MB0F0     LL       .000000       .000000       .008438       .000000
```

.

.

.

(continues with other row sections)

APPENDIX E
PRODUCTION ACCOUNTING MODULE

CONTENTS

Table E-1

File: VERHOR.GSM (Location: \GSAM\PRODACCT\DATA)
This file is output from the horizontal / vertical selection routine in the Exploration and Production Module. It lists the vertical /horizontal decision made for each undiscovered or undeveloped reservoir (GSAMID and counter; “0” for vertical and “1” for horizontal. This file has been shortened to fit to one page in the Appendix.

```
01116701F005   0  Vertical Well
01116701F006   0  Vertical Well
01116701F007   0  Vertical Well
01116701F008   0  Vertical Well
01116701F009   0  Vertical Well
01116701F010   1  Horizontal Well
01116701F011   1  Horizontal Well
01116701F012   1  Horizontal Well
01116701F013   1  Horizontal Well
01116701F014   0  Vertical Well
01116701F015   0  Vertical Well
01116701F016   0  Vertical Well
01116701F017   0  Vertical Well
01116702F005   0  Vertical Well
01116702F006   0  Vertical Well
01116702F007   0  Vertical Well
01116702F008   0  Vertical Well
01116702F009   0  Vertical Well
01116702F010   1  Horizontal Well
01116702F011   1  Horizontal Well
01116702F012   1  Horizontal Well
01116702F013   1  Horizontal Well
01116702F014   0  Vertical Well
01116702F015   0  Vertical Well
01116702F016   0  Vertical Well
01116702F017   0  Vertical Well
01116703F005   0  Vertical Well
01116703F006   0  Vertical Well
01116703F007   0  Vertical Well
01116703F008   0  Vertical Well
01116703F009   0  Vertical Well
01116703F010   1  Horizontal Well
01116703F011   1  Horizontal Well
01116703F012   1  Horizontal Well
01116703F013   1  Horizontal Well
01116703F014   0  Vertical Well
01116703F015   0  Vertical Well
01116703F016   0  Vertical Well
01116703F017   0  Vertical Well
01116704F005   0  Vertical Well
```

.

.

.

(continues with other reservoirs)

Table E-2

File: OUTPUT.OPT (Location: \GSAM\PRODACCT)

```
Write the Pro-Forma for Every Reservoir ( 1:YES, 0:NO, file:reservoir.out)
0
Write the Pro-Forma for the STATE/REGION (1:STATE, 0:REGION, file:state.out/region.out )
0
Write the Pro-Forma for the entire Nation ( 1:YES, 0:NO, file:nat.out )
1
Enter the Identifiers for Resource Types For Which Pro-Forma is Requested (file:res.out)
1 2 3 4 5 6 7
8606    \ Number of Undiscovered Reservoirs Units
1997    \ Last Year For Which Data is Available (1997 currently)
2020    \ End Year of Analysis
2021    \ Year Env. Regs Are Effective (Used Only When Regulations Specified in RP Module)
0       \ Enter 1 for federal only run, otherwise enter 0
```

(Value superceded by ENV_DAT.SPC entries)

Table E-3

File: COST.DAT (Location: \ GSAM\PRODACCT\DATA)
This file should not be confused with the file of the same name from the Reservoir Performance Module. The RP module uses multiple costing files to represent all of the regions, which the model deals with. Conversely, the P&A module reads one costing file. Hence, the RP module should be "run" using multiple cost files (by copying appropriate costing file in generic COST.DAT file corresponding to the .GSM file). The P&A module only takes one cost file and correctly represents regional and horizontal/vertical costs.

```
.
C*** Discount  Rate(%)
10.0
C*** Number of Cases
2
C*** NAME OF TECHNOLOGY ONE
Current Technology
C*** Number of Regions For G&G & Lease bonus Factors (Portion of EWC that is G&G; Lease Bonus)
3
C**  G&G    Lease Bonus
1    .05    .01
16   .05    .01
5    .05    .01
C*** Default
99   .05    .01
C*** Number of Regions For % Dev.Well,Dry Infill,Succ. Infill Costs; % Tang. Exp.Cost, Tang. Dev.Cost,
Tang. Facilities Cost
2
C* % Dev.Well    Dry Inf.  Suc.Inf.   % Tang.Exp.   % Tang.Dev.  % Tang.Facilities
6    70.0         70.0      70.0       25.0          40.0         100.0
8    70.0         70.0      70.0       25.0          40.0         100.0
C*** Default
99   70.0         70.0      70.0       25.0          40.0         100.0
C*** Number of Regions For Env.Cost Mult.(Function of Facilities), G&A Expense Mult;, and G&A Capital
Mult.
1
C*** Env. Cap. Cost. Mult.    G&A Expense Mult.   G&A Capital Mult.
8           0.10                  0.25              0.10
C***
99          0.10                  0.25              0.10
C*** Number of Regions For Development Well Cost (Thousand Dollars)
22
Region#  Intercept    X-Coeff.      X2-Coeff.     X3-Coeff.     Vert. Factor   Horz. Factor
  1      27.0688     4.7098399e-2 -2.547277e-6  1.18087525e-10      1          1.3
  2      59.2575     5.948449e-2  -3.611307e-6  3.975062e-10        1          1.3
  3      34.1655     9.042406e-2  -3.209655e-7  9.641927e-10        1          1.3
  4      46.5565    -1.905254e-2   1.430119e-5 -3.249473e-10        1          1.3
  5      299.090     1.280414e-1  -1.890103e-5  1.784502e-9         1          1.3
  6      21.7314     1.848788e-3   8.429423e-6  2.456291e-10        1          1.3
  7      35.6535     8.322879e-2  -1.829027e-5  1.632399e-9         1          1.3
  8      78.0708     1.775666e-2   1.481531e-6  3.708914e-10        1          1.3
  9      54.4073     5.387449e-2  -7.106254e-6  1.339750e-9         1          1.3
 10      47.2167     5.970161e-2  -6.851103e-6  7.622377e-10        1          1.3
 11      17.8909     3.34051e-2    1.13753e-6   0.0                 1          1.3
 12      200.000    -2.418747e-2   2.630253e-5 -6.552413e-10        1          1.3
 13      3431.02     7.761334e-1  -5.736602e-5  3.112882e-9         1          1.3
 14      1715.51     3.880667e-1  -2.868301e-5  1.556441e-9         1          1.3
 15      1715.51     3.880667e-1  -2.868301e-5  1.556441e-9         1          1.3
 16      1715.51     3.880667e-1  -2.868301e-5  1.556441e-9         1          1.3
 17      2239.81    -4.69329e-2    2.14607e-5   3.03929e-10         1          1.3
 18      1000.00     1.848788e-3   8.429423e-6  2.456291e-10        1          1.3
 19      300.000     1.848788e-3   8.429423e-6  2.456291e-10        1          1.3
 22      75.0000     5.63445e-3    1.34339e-5   0.000               1          1.5
 23      97.5000     7.32479e-3    1.74607e-5   0.000               1          1.5
 24      8000.00     100.200e-3    0.000        0.000               1          1.3
C*** Default Data for Development Well Cost (Thousand Dollars)
 99      78.07079    1.775666e-2   1.481531e-6  3.708914e-10        1          1.3
C**Number of Regions For Incremental Environmental Costs (GSAM Env. Module)
0
Region# Ex.Tan. Ex.Int. Ex. O&M(C) New Tan.(C) New Int.(C) New O&M(C) Env.$/Ft  Env.$/MCF   Env.$/Bbl
  99      0       0        0          0           0            0        0                    0
0
C*** Number of Regions For Facilities Well Cost (Function of flow potential MCF/D)
```

```
2
Region#  # of Depth Steps
01             1
Max Depth      $/Well   $/Well/MCF-D
---------    --------- --------------
160000         261.8       26.18
Region#  # of Depth Steps
08             5
Max Depth      $/Well   $/Well/MCF-D
---------    --------- --------------
2000          25673.64      11.50
4000          27342.41      11.29
8000          37308.32      10.06
12000         43224.96       9.35
16000         51549.09       8.27
Region#  # of Depth Steps
99             5
Max Depth      $/Well   $/Well/MCF-D
---------    --------- --------------
2000          30268.50      10.96
4000          35788.26      10.27
8000          38474.28       9.92
12000         44009.35       9.25
16000         50790.63       8.35
C*** Number of Regions For Stimulation Efficiency (STMFAC Value), Fraction
1
Region#   Stimulation Efficiency (STMFAC Value)
10                0.60
C*** Default Data For Stimulation Efficiency (STMFAC Value), Fraction
99                0.60
C*** Number of Regions For Compressor Costs ($/BHP), Var. O&M ($/MCF)
3
Region#  Comp.Cost  Var. O&M  Incr. per 1000 ft.  Water O&M ($/Bbl)   Comp. O&M
22         1200      0.005          0.0              0.25             0.15
23         1200      0.005          0.0              0.25             0.15
24         1200      0.005          0.0              0.25             0.15
C*** Default Data For Compressor Costs and Var. O&M
99         1200      0.005          0.0              0.25             0.05
C*** Annual Fixed O&M Well Cost (function of well depth)
C*** Number of Regions (Excluding Default - 99)
12
Region#    Number of Steps
01               1                      (Appalachia)
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         1003.1        0.40
Region#    Number of Steps
04               1                       (Arkla-East Texas)
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         7514.0        1.69
Region#    Number of Steps
05               1                       (So-Louisiana)
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         8040.0        1.66
Region#    Number of Steps
06               1                       (Texas Gulf Coast)
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         7318.0        1.96
Region#    Number of Steps
07                1                       (Permian)
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         6211.0        1.49
Region#    Number of Steps
08                1                       (Mid-Continent)
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         8609.0        1.56
Region#    Number of Steps
10                1                       (Rockies Foreland)
Max. Depth       $/Well      $/(Well-ft)
```

```
C---------     C---------    C----------
 15000.0         8963.0       1.04
C-  C-    Region and Number of Steps
13  1                                       (Atlantic Offshore)
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0       298786.0        0.00
C-  C-    Region and Number of Steps
14  1                                       (Gulf of Mexico-East)
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0       298786.0        0.00
Region#     Number of Steps
15                    1                    (Norphlet)
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0       298786.0        0.00
Region#     Number of Steps
16                    1                     (Gulf of Mexico-Cntr)
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0       250757.0        0.00
Region#     Number of Steps
17                    1                     (Gulf of Mexico-West)
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0       250757.0        0.00
Region#     Number of Steps
99                    1                     (Default)
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         8869.0        1.44
```

This is the part of the file that specifies the costs with Current Technology used. The actual file also contains cost specifications for Advanced Technology used.

Table E-4

File: DRILLADJ.DAT (Location: \GSAM\PRODACCT\DATA)
This file contains the adjustments to GSAM exploration and development well drilling in the PA module for cash flow purposes. This file is used to calibrate GSAM wells to JAS wells by well type (exploration as well as development wells). Wells are calibrated for a specified year and are adjusted for every year thereafter.

```
1995   ACTUAL          Dev. Wells    Exp. Wells

Pacific Offshore              0         0
Pacific Onshore               0         0
San Juan                      0         0
Rockies Foreland              0         0
Williston                     0         0
Permian                       0         0
Mid-Continent                 0         0
Arkla-East Texas              0         0
Texas Gulf Coast              0         0
Gulf of Mexico-West           0         0
Gulf of Mexico-Cntr           0         0
Norphlet                      0         0
Gulf of Mexico-East           0         0
So-Louisiana                  0         0
MAFLA Onshore                 0         0
Mid-West                      0         0
Appalachia                    0         0
Alberta                       0         0
British Columbia              0         0
North Alaska                  0         0
MacKenzie Delta               0         0
Atlantic Offshore             0         0
Mexico Supply                 0         0
```

The adjustment year is specified at the top left of DRILLADJ.DAT file with well calibration numbers below. If the numbers in file are zero then there are no adjustments to the number of GSAM wells drilled. DRILLADJ.JAS contains the actual number of wells drilled in year 1995 according to JAS survey. If DRILLADJ.JAS file is used in the PA run (by copying it into DRILLADJ.DAT) then in year 1995 the PA wells would be exactly the same as the number specified in DRILLADJ.JAS file. For example; if number of development wells drilled in Texas Gulf Coast in 1995 in EP module are 1200, then the PA module will increase it to 1234 (as specified in DRILLADJ.JAS file). So, cashflow associated with extra 1234-1200 = 34 wells is also accounted in pro-forma cashflow in the PA module. In addition, 34 wells get added in every year from 1995 onwards in the Texas Gulf Coast region. In addition, if in Rockies Foreland region if EP module drills 850 development wells in 1995; then since 817-850 = -33 is a negative number, no wells are added in Rockies Foreland in year 1995 and no adjustments are made thereafter.

Table E-5

File: TAX_NAT.DAT (Location: \GSAM\PRODACCT\DATA)
This national tax file must be consistent with that in the Reservoir Performance Module. See Appendix Table B-7 for a complete description.

```
C*** U.S. Federal Income Tax Rate
34.0
C*** Canadian Federal Income Tax Rate
28.0
C*** Independent Producer Depletion Rate (%)
100.0
C***  Are Intangible Drilling Costs to be Capitalized? (YES/NO)
YES
C***  Are Other Intangibles to be Capitalized? (YES/NO)
YES
C***  Include environmental Costs? (YES/NO)
YES
C***  Are Environmentals to be Capitalized? (YES/NO)
NO
C***  Implement Alternative Minimum Taxes? (YES/NO)
NO
C***  Allow AMT Taxes Paid to be Used as Credits in Future Years? (YES/NO)
YES
C***  Six Month Amortization Rate (%)
50.0
C***  Intangible Drilling Cost Preference Deduction (%)
100.0
C***  ACE Rate (%)
70.0
C***  Maximum Alternative Minimum Tax Reduction for Independents
0.0
C***  Alternative Minimum Tax RATE (%)
20.0
C***  Expense Environmental Costs? (YES/NO)
NO
C***  Allow Net Income Limitations? (YES/NO)
NO
C***  Net Income Limitation Limit (%)
40.0
C***  Percent Depletion Rate (%)
0.0
C***  Percent of Intan. Inv. to Capitalize (%)
30.0
C***  EOR Tax Credit Rate (%)
15.0
C***  Allow G&G Depletable Tax Credit? (YES/NO)
NO
C***  G&G Depletable Tax Credit Rate (%)
10.0
C***  Allow Tax Credit for Expensed G&G? (YES/NO)
NO
C***  G&G Intangible Tax Credit Rate (%)
15.0
C***  Allow Lease Acq. Depletable Tax Credit? (YES/NO)
NO
C***  Lease Acq. Depletable Tax Credit Rate (%)
```

```
10.0
C***  Allow Tax Credit for Expensed Lease Acq. Costs? (YES/NO)
NO
C***  Tax Credit Rate for Expensed Lease Acq. Costs (%)
15.0
C***  Allow Tangible Development Tax Credit? (YES/NO)
NO
C***  Tangible Development Tax Credit Rate (%)
15.0
C***  Allow Intangible Drilling Cost Tax Credit? (YES/NO)
NO
C***  Intangible Drilling Cost Tax Credit Rate (%)
15.0
C***  Allow Other Intangible Tax Credit? (YES/NO)
NO
C***  Other Intangible Tax Credit Rate (%)
15.0
C***  Allow Environmental Tangible Tax Credit? (YES/NO)
NO
C***  Environmental Tangible Tax Credit Rate (%)
20.0
C***  Allow Environmental Intangible Tax Credit? (YES/NO)
NO
C***  Environmental Intangible Tax Credit Rate (%)
20.0
C***  Allow Environmental Operating Cost Tax Credit? (YES/NO)
NO
C***  Environmental Operating Cost Tax Credit Rate (%)
20.0
C***  Allow Tax Credit On Tangible Investments? (YES/NO)
NO
C***  Number of Years for Tax Credit on Tangible Investments
20
C***  Allow Tax Credit On Intangible Investments? (YES/NO)
NO
C***  Number of Years for Tax Credit on Intangible Investments
15
C***  Percent of G&G Depleted (%)
16.17
C***  Allow  Forgiveness of State Taxes? (YES/NO)
NO
C***  Number of Years for Forgiveness of State Taxes
10
C***  Percent Lease Acquisition Cost Capitalized
100.0
```

Table E-6

File: TAXES.DAT (Location: \GSAM\PRODACCT\DATA)
This file must be consistent with that in the Reservoir Performance Module. See Appendix Table B-8 for a complete description.

```
C*** State Tax Rates - Oil Severance Rates - Gas Severance Rates
C*** Number of Regions (Excluding Default - 99)
60
Code     State   Oil     Oil    Gas      Gas     Ad-Valorem
C*        (%)    (%)   ($/Bbl)  (%)    ($/MCF)  Tax (% of prod)
0100     0.00   10.00   0.000  10.00   0.00000   0.00         :Code| State Tax| Oil Sev. Tax| Gas Sev. Tax|
0105     5.00   10.00   0.000  10.00   0.00000   0.00         :0100:ALABAMA FED.OFFSHORE,0105:STATE OFFSHORE
0110     5.00   10.00   0.000  10.00   0.00000   0.00         :ALABAMA ONSHORE              :             :
5000     0.00   15.00   0.004  10.00   0.00008   0.00         :ALASKA SOUTH FED. OFFSHORE   :             :
5005     9.40   15.00   0.004  10.00   0.00008   0.00         :ALASKA SOUTH STATE OFFSHORE  :             :
5010     9.40   15.00   0.004  10.00   0.00008   0.00         :ALASKA SOUTH ONSHORE         :             :
5050     9.40   15.00   0.004  10.00   0.00008   0.00         :ALASKA NORTH ONSHORE         :             :
2        9.00    0.00   0.000   0.00   0.00000   0.00         :ARIZONA                      :             :
0310     6.50    5.00   0.000   0.00   0.00300   0.00         :ARKANSAS SOUTH               :             :
0350     6.50    5.00   0.000   0.00   0.00300   0.00         :ARKANSAS NORTH               :             :
36       10.0    5.00   0.000   5.00   0.00000   0.00         :OREGON                       :             :
0400     0.00    0.00   0.025   0.00   0.02500   0.00         :CALIFORNIA FED. OFFSHORE     :             :
0405     9.30    0.00   0.025   0.00   0.02500   0.00         :CALIFORNIA STATE OFFSHORE    :             :
0410     9.30    0.00   0.025   0.00   0.02500   0.00         :CALIFORNIA CENTRAL VALLEY    :             :
0450     9.30    0.00   0.025   0.00   0.02500   0.00         :CALIFORNIA COASTAL           :             :
0490     9.30    0.00   0.025   0.00   0.02500   0.00         :CALIFORNIA LOS ANGELES BASIN :             :
5        5.00    5.17   0.000   5.17   0.00000   0.00         :COLORADO                     :             :
0900     0.00    8.00   0.000   0.00   0.12400   0.00         :FLORIDA FED. OFFSHORE        :             :
0910     5.50    8.00   0.000   0.00   0.12400   0.00         :FLORIDA ONSHORE              :             :
12       4.80    0.00   0.000   0.00   0.00000   0.00         :ILLINOIS                     :             :
13       4.50    1.00   0.000   0.00   0.00000   0.00         :INDIANA                      :             :
15       4.00    8.00   0.000   8.00   0.00000   0.00         :KANSAS                       :             :
1700     0.00   12.50   0.000   0.00   0.07000   0.00         :LOUISIANA FED. OFFSHORE      :             :
1705     8.00   12.50   0.000   0.00   0.07000   0.00         :LOUISIANA STATE OFFSHORE     :             :
1710     8.00   12.50   0.000   0.00   0.07000   0.00         :LOUISIANA SOUTH              :             :
1750     8.00   12.50   0.000   0.00   0.07000   0.00         :LOUISIANA NORTH              :             :
21       2.30    6.60   0.000   5.00   0.00000   0.00         :MICHIGAN                     :             :
2300     0.00    6.00   0.000   6.00   0.00000   0.00         :MISSISSIPPI FED. OFFSHORE    :             :
2310     5.00    6.00   0.000   6.00   0.00000   0.00         :MISSISSIPPI ONSHORE          :             :
25       6.75    5.00   0.000   2.65   0.00000   0.00         :MONTANA                      :             :
26       7.81    3.00   0.000   3.00   0.00000   0.00         :NEBRASKA                     :             :
3010     7.60    7.09   0.000   9.755  0.00000   0.00         :NEW MEXICO SOUTHEAST         :             :
3050     7.60    7.09   0.000   9.755  0.00000   0.00         :NEW MEXICO NORTHWEST         :             :
33      10.50    5.00   0.000   2.00   0.00000   0.00         :NORTH DAKOTA                 :             :
3510     6.00    7.00   0.000   7.00   0.00000   0.00         :OKLAHOMA SOUTHWEST           :             :
3520     6.00    7.00   0.000   7.00   0.00000   0.00         :OKLAHOMA SOUTHEAST           :             :
3530     6.00    7.00   0.000   7.00   0.00000   0.00         :OKLAHOMA NORTHEAST           :             :
3540     6.00    7.00   0.000   7.00   0.00000   0.00         :OKLAHOMA NORTH CENTRAL       :             :
3550     6.00    7.00   0.000   7.00   0.00000   0.00         :OKLAHOMA NORTHWEST           :             :
40       6.00    4.50   0.000   4.50   0.00000   0.00         :SOUTH DAKOTA                 :             :
4200     0.00    4.60   0.000   7.50   0.00000   0.00         :TEXAS FED. OFFSHORE          :             :
4205     0.00    4.60   0.000   7.50   0.00000   0.00         :TEXAS STATE OFFSHORE         :             :
4210     0.00    4.60   0.000   7.50   0.00000   0.00         :TEXAS RRC DISTRICT 1         :             :
4220     0.00    4.60   0.000   7.50   0.00000   0.00         :TEXAS RRC DISTRICT 2         :             :
4230     0.00    4.60   0.000   7.50   0.00000   0.00         :TEXAS RRC DISTRICT 3         :             :
4240     0.00    4.60   0.000   7.50   0.00000   0.00         :TEXAS RRC DISTRICT 4         :             :
4250     0.00    4.60   0.000   7.50   0.00000   0.00         :TEXAS RRC DISTRICT 5         :             :
4260     0.00    4.60   0.000   7.50   0.00000   0.00         :TEXAS RRC DISTRICT 6         :             :
4270     0.00    4.60   0.000   7.50   0.00000   0.00         :TEXAS RRC DISTRICT 7B        :             :
4275     0.00    4.60   0.000   7.50   0.00000   0.00         :TEXAS RRC DISTRCIT 7C        :             :
4280     0.00    4.60   0.000   7.50   0.00000   0.00         :TEXAS RRC DISTRICT 8         :             :
4285     0.00    4.60   0.000   7.50   0.00000   0.00         :TEXAS RRC DISTRICT 8A        :             :
4290     0.00    4.60   0.000   7.50   0.00000   0.00         :TEXAS RRC DISTRICT 9         :             :
4295     0.00    4.60   0.000   7.50   0.00000   0.00         :TEXAS RRC DISTRICT 10        :             :
43       5.00    5.00   0.000   5.00   0.00000   0.00         :UTAH                         :             :
47       9.00    5.00   0.000   5.00   0.00000   0.00         :WEST VIRGINIA                :             :
49       0.00    5.06   0.000   5.06   0.00000   7.662        :WYOMING                      :             :
5300    15.50    4.50   0.000   4.50   0.00000   0.00         :CANADA  (Alberta)            :
5301    16.50    9.00   0.000   9.00   0.00000   0.00         :CANADA  (British Columbia)   :
5302    17.00   10.00   0.000  10.00   0.00000   0.00         :CANADA  (Saskatchewan)       :
9900     10.0    5.00   0.000   5.00   0.00000   0.00         :DEFAULT RATES                :             :
```

Table E-7

Input Data File: TECH.DAT (Location: \GSAM\PRODACCT\DATA)
These technology file must be consistent with this in the Reservoir Performance Module. See Appendix Table B-9 for a complete description. Here the Current Technology section is shown only.

```
C*** Number of Technologies
2
C*** Name of Technology One
Current Technology
C*** Dry Hole Probability (%)
20.0
C*** Year to drill Infill wells for Water Drive Reservoirs
5.0
c**** number of regions for proration
6
c  Region Number and Proration (Fraction)
1   0.07
13  0.25
14  0.25
15  0.25
16  0.25
17  0.25
c   Default proration factor
99  0.10
c****number of states for state specific proration
0
c proration factors by state
c**** Number of different regions for Pay Continuity Enhancement
1
c  Pay Enhancement (Based on BEG study)
1  1.0
c  Default for Pay Enhancement
99  1.0
c**** Number of different regions for System Pressure
1
c  Minimum system pressures by region
1    20.
c  Default for Minimum System Pressure
99  150.
c  Number of Reservoir Types to Describe Well Performance Factors
6
c  Vertical Well Skin Factors for Reservoir Types 1 through Number above (For Vertical Well,
Horz Skin calculated in the model)
15.    12.    13.    13.    15.    15.
c  Well Radius for Reservoir Types 1 through Number Above (Assume 9 inch hole)
0.354  0.354  0.354  0.354  0.354  0.354
c  Fracture Half Lengths for Reservoir Types 1 through Number above
0.     300.   300.   300.    0.    150.
c  Fracture Conductivity for Reservoir Types 1 through Number above
0.      100.   100.   100.   0.      50.
c number of regions for horizontal wells
0
c enter horizontal well info
c***** Number of different regions for tubing diameter
1
c ****** Enter tubing size by region (inches) (Assume 2 7/8 tubing)
1   1.4
c ****** Enter tubing size default (inches)
99  1.995
c end of technology
```

Table E-8

OUTPUT File: NAT.OUT (Location: \GSAM\PRODACCT)
Contains U.S. non-associated gas production and related pro-forma entries. The width of the file has been abridged to fit the page; the actual file contains data for the years 1997-2020.

```
Detailed Financial Report
     ========= National Total ===========


                                  Year         1997      1998      1999      2000      2001
                                             ========  ========  ========  ========  ========
Total Gas Production (TCF)                     16.606    16.260    16.652    17.027    17.428
Total Federal Gas Production (TCF)              5.577     5.610     5.662     5.661     5.614
Total Liquids Production (Bil Bbl)            616.183   592.082   577.099   564.536   547.734
Average Gas Price ($/MCF)                        2.13      1.87      2.01      2.05      2.09
Average Oil Price ($/Bbl)                         .00       .00       .00       .00       .00
Total Gross Revenues (Billion $)               35.363    30.471    33.436    34.873    36.406
 Total Gravity/Trans. Cost Adj.                  .000      .000      .000      .000      .000
Total Adjusted Revenues                        35.363    30.471    33.436    34.873    36.406
 Total Royalties                                4.420     3.809     4.180     4.359     4.551
 Total Federal Royalties                        1.475     1.355     1.455     1.440     1.409
 Total Private Royalties                        2.945     2.454     2.725     2.919     3.142
Total Net Sales                                30.942    26.662    29.257    30.514    31.855
Total Operating Cost                            6.725     5.860     5.798     5.661     5.540
 Total G&A on Expensed Items (on Gen.O&M)       1.079      .992      .975      .932      .903
 Total G&A on Capitalized Items (on tot.inv.     .883      .392      .432      .490      .492
 Total Pressure Maint./Cycling                   .000      .000      .000      .000      .000
 Total General O&M (Fixed,Var.,Comp.)           4.317     3.962     4.012     3.952     3.938
 Total Processing O&M                            .407      .382      .381      .389      .394
 Total Environmental O&M Costs                   .000      .007     -.113     -.222     -.328
 Total Stimulation Costs                         .039      .126      .111      .120      .141
 Total Recompletion Costs                        .000      .000      .000      .000      .000
Total Intangible Investment                     1.785     2.619     2.844     3.193     3.094
 Intang. Exploratory Costs                      1.417     1.241     1.342     1.607     1.407
 Intang. Development Costs                       .368      .800      .835      .897     1.035
 Intang. Environmental Costs                     .000      .578      .667      .690      .652
 Other Intangible Costs                          .000      .000      .000      .000      .000
Portion of Intangibles to Capitalize             .535      .612      .653      .751      .733
TOTAL INVESTMENTS                               8.825     3.917     4.319     4.896     4.918
Tangible Investments                            7.040     1.298     1.475     1.703     1.824
 Tang. Exploratory Cost                          .468      .414      .445      .533      .467
 Tang. Development Cost                          .243      .535      .555      .596      .686
 Tang. Environmental Cost                        .032      .002      .002      .003      .003
 Other Tang. Cost                               6.297      .347      .473      .572      .668
Total Depreciable/Capitalized Investments       7.576     1.910     2.128     2.454     2.557
 Adj. for Federal Tax Credits                    .000      .000      .000      .000      .000
Adj. Depreciable/Capitalize Inv                 7.576     1.910     2.128     2.454     2.557
Depreciation                                    1.269     2.131     2.093     2.135     2.238
Depletable G&G/Lease Acq. Costs                 1.900      .980      .415      .320      .262
 Depletable Lease Acq. Cost                     1.884      .966      .400      .302      .246
 Depletable G&G Costs                            .016      .014      .015      .018      .016
 Adjustments for Federal Tax Credits             .000      .000      .000      .000      .000
Depletion Base                                  1.900      .980      .415      .320      .262
Expensed G&G/Lease Acq. Costs                    .084      .073      .079      .095      .083
 Expensed Lease Acq. Cost                        .000      .000      .000      .000      .000
 Expensed G&G Costs                              .084      .073      .079      .095      .083
Operating Cost/mcf                               .405      .360      .348      .332      .318

 Industry Jobs (Thou. - Rev Based)               113.       98.      107.      112.      116.
 Industry Jobs (Thou. - Cost Based)              148.       93.       96.      100.       99.

 Total Jobs (Thou. - Rev Based)                  318.      274.      301.      314.      328.
 Total Jobs (Thou. - Cost Based)                 415.      261.      270.      282.      279.
```

Table E-8 (continued)

OUTPUT File: NAT.OUT

```
Detailed Financial Report
     ========= National Total ===========

                                  Year         1997      1998      1999      2000      2001
                                             ========  ========  ========  ========  ========
Total Operating Wells                         160773.   134045.   139452.   137435.   138594.
  Exploratory Wells (Incl. Dry Holes)           1315.     1135.     1360.     1412.     1327.
  Total Primary Development Wells                976.     2123.     2886.     3398.     3998.
   - Successful Development Wells                877.     1910.     2597.     3058.     3597.
   - Dry Development Wells                        98.      213.      289.      340.      400.
  Total Infill Wells                               0.      714.      154.       95.      143.
   - Successful Infill Wells                       0.      571.      124.       76.      115.
   - Dry Infill Wells                              0.      143.       31.       19.       28.
  Total Dry Wells Drilled (Development)           98.      356.      320.      359.      429.

  Total Wells (Dev. & Expl.)                    2291.     3972.     4401.     4905.     5468.
Net Revenues (Billion $)                      30.942    26.662    29.257    30.514    31.855
 Operator Severance Taxes                      1.929     1.679     1.837     1.936     2.047
 Operating Costs                               6.725     5.860     5.798     5.661     5.540
 Expensed Int.,G&G, and Lease Acq.             1.333     2.080     2.270     2.537     2.444
 Depreciation                                  1.269     2.131     2.093     2.135     2.238
 Depletion Allowance                            .008      .022      .040      .060      .081
Taxable Income                                19.679    14.889    17.218    18.186    19.505
 Tax Credit Addback                             .000      .000      .000      .000      .000
 Intangible Addback                             .000      .000      .000      .000      .000
 G&G/Lease Addback                              .000      .000      .000      .000      .000
Net Income Before Taxes                       19.679    14.889    17.218    18.186    19.505
 State Income Taxes                             .383      .227      .271      .271      .317
 Federal Income Tax                            6.561     4.985     5.762     6.091     6.524
 Federal Tax Credits                            .000      .000      .000      .000      .000
Net Income After Taxes                        12.735     9.677    11.185    11.824    12.665
 plus Depreciation                             1.269     2.131     2.093     2.135     2.238
 plus Depletion                                 .008      .022      .040      .060      .081
 less Depletable Items                         1.900      .980      .415      .320      .262
 less Depreciable/Capitalized Items            7.576     1.910     2.128     2.454     2.557
 less Tax Credit on Expensable Items            .000      .000      .000      .000      .000
Annual After Tax Cash Flow                     4.536     8.940    10.776    11.245    12.164


Total CO2 Production (BCF)                   167.652   160.888   158.545   153.990   154.159
Total Nitrogen Production (BCF)              239.252   224.410   228.196   232.743   232.478
Total Hydro-Sulf Production (k ton)         1547.320  1414.912  1357.560  1347.375  1344.225
```

Table E-9

OUTPUT File: REGION.OUT (Location: \GSAM\PRODACCT)
Contains non-associated gas production and related pro-forma entries for each region in the model. The width of the file has been abridged to fit the page; the actual file contains data for the years 1997-2020. A sample region entry is shown below.

```
Detailed Financial Report
     GSAM Region:  1 Case.: Current Technology  Region Name = Appalachia


                                  Year          1997      1998      1999      2000      2001      2002      2003      2004      2005      2006      2007      2008      2009      2010
                                             ========  ========  ========  ========  ========  ========  ========  ========  ========  ========  ========  ========  ========  ========
Total Gas Production (BCF)                       .000      .000      .000      .000      .000      .000     1.013     2.018     3.002     4.665     6.337     8.030     9.805    11.614
Total Fed. Gas Production (BCF)                  .000      .000      .000      .000      .000      .000     1.013     2.018     3.002     4.665     6.337     8.030     9.805    11.614
Gas Price ($/MCF)                                2.56      2.04      2.08      2.02      2.07      2.15      2.27      2.36      2.41      2.41      2.39      2.38      2.39      2.43
Total Liquids Production (Bil Bbl)               .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000
Oil Price ($/Bbl)                                 .00       .00       .00       .00       .00       .00       .00       .00       .00       .00       .00       .00       .00       .00
Total Gross Revenues (MM$)                       .000      .000      .000      .000      .000      .000     2.299     4.762     7.235    11.243    15.145    19.112    23.433    28.222
 Total Gravity/Trans. Cost Adj.                  .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000
Total Adjusted Revenues                          .000      .000      .000      .000      .000      .000     2.299     4.762     7.235    11.243    15.145    19.112    23.433    28.222
 Total Royalties                                 .000      .000      .000      .000      .000      .000      .287      .595      .904     1.405     1.893     2.389     2.929     3.528
 Total Federal Royalties                         .000      .000      .000      .000      .000      .000      .287      .595      .904     1.405     1.893     2.389     2.929     3.528
 Total Private Royalties                         .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000
Total Net Sales                                  .000      .000      .000      .000      .000      .000     2.012     4.166     6.331     9.838    13.252    16.723    20.504    24.694
Total Operating Cost                             .020      .013      .011      .009      .007      .014      .937      .993     1.088     1.895     2.033     2.181     2.317     2.580
 Total G&A on Expensed Items (on Gen. O&M)       .000      .000      .000      .000      .000      .000      .016      .026      .030      .066      .100      .133      .166      .201
 Total G&A on Capitalized Items (on tot.inv.     .020      .013      .011      .009      .007      .014      .236      .252      .337      .548      .549      .563      .559      .593
 Total Pressure Maint./Cycling                   .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000
 Total General O&M (Fixed,Var.,Comp.)            .000      .000      .000      .000      .000      .000      .089      .179      .269      .429      .588      .749      .916     1.093
 Total Processing O&M                            .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000
 Total Environmental O&M Costs                   .000      .000      .000      .000      .000      .000     -.026     -.077     -.149     -.166     -.188     -.217     -.250     -.290
 Total Stimulation Costs                         .000      .000      .000      .000      .000      .000      .622      .613      .600     1.019      .985      .953      .927      .983
 Total Recompletion Costs                        .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000
Total Intangible Investment                      .148      .094      .079      .062      .052      .101     1.320     1.462     2.140     3.214     3.275     3.462     3.455     3.718
 Intang. Exploratory Costs                       .148      .094      .079      .062      .052      .101      .000      .062      .633      .039      .030      .024      .049      .000
 Intang. Development Costs                       .000      .000      .000      .000      .000      .000      .877      .865      .847     2.200     2.126     2.058     2.001     2.093
 Env. Intangible Capital Costs                   .000      .000      .000      .000      .000      .000      .443      .535      .660      .975     1.118     1.380     1.405     1.625
 Other Intangible Costs                          .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000
Portion of Intangibles to Capitalize             .044      .028      .024      .019      .016      .030      .263      .278      .444      .672      .647      .625      .615      .628
TOTAL INVESTMENTS                                .196      .129      .109      .086      .072      .137     2.360     2.519     3.373     5.476     5.485     5.626     5.589     5.932
Tangible Investments                             .048      .035      .030      .024      .021      .037     1.040     1.057     1.233     2.263     2.210     2.163     2.134     2.214
 Tang. Exploratory Cost                          .048      .031      .026      .021      .017      .033      .000      .020      .207      .013      .010      .008      .016      .000
 Tang. Development Cost                          .000      .000      .000      .000      .000      .000      .577      .567      .554     1.439     1.392     1.348     1.310     1.367
 Tang. Environmental Cost                        .000      .003      .003      .003      .003      .003      .004      .004      .004      .004      .004      .004      .004      .004
 Other Tangible Cost                             .000      .000      .000      .000      .000      .000      .459      .465      .469      .807      .804      .803      .804      .842
Total Depreciable/Capitalized Investments        .092      .063      .053      .042      .036      .067     1.303     1.335     1.677     2.934     2.857     2.788     2.749     2.842
 Adj. for Federal Tax Credits                    .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000
Adj. Depreciable/Capitalize Inv.                 .092      .063      .053      .042      .036      .067     1.303     1.335     1.677     2.934     2.857     2.788     2.749     2.842
Depreciation                                     .013      .032      .039      .042      .041      .046      .232      .544      .817     1.240     1.713     2.062     2.329     2.511
Depletable G&G/Lease Acq. Costs                  .002      .001      .001      .001      .001      .001     5.735      .001      .007     1.875      .000      .000      .001      .166
 Depletable Lease Acq. Cost                      .000      .000      .000      .000      .000      .000     5.735      .000      .000     1.874      .000      .000      .000      .166
 Depletable G&G Costs                            .002      .001      .001      .001      .001      .001      .000      .001      .007      .000      .000      .000      .001      .000
 Adjustments for Federal Tax Credits             .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000
Depletion Base                                   .002      .001      .001      .001      .001      .001     5.735      .001      .007     1.875      .000      .000      .001      .166
Expensed G&G/Lease Acq. Costs                    .009      .006      .005      .004      .003      .006      .000      .004      .037      .002      .002      .001      .003      .000
 Expensed Lease Acq. Cost                        .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000      .000
 Expensed G&G Costs                              .009      .006      .005      .004      .003      .006      .000      .004      .037      .002      .002      .001      .003      .000
Operating Cost/Mcf                               .000      .000      .000      .000      .000      .000      .925      .492      .362      .406      .321      .272      .236      .222

 Industry Jobs (Thou.- Rev Based)                  0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.
 Industry Jobs (Thou. - Cost Based)                0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.

 Total Jobs (Thou. - Rev Based)                    0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.
 Total Jobs (Thou. - Cost Based)                   0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0.        0..
```

Table E-10

OUTPUT File: RES.OUT (Location: \GSAM\PRODACCT)
Contains non-associated gas production and the related pro-forma entries by resource type. The width of the file has been abridged to fit the page; the actual file contains data for the years 1997-2020. Here the section for Conventional Gas is shown only. The rest of the resource sections have identical format.

```
Detailed Financial Report
     ========= Resource Aggregations ===========

 ========== Conventional Gas Resource              =============
                                  Year           1997      1998      1999      2000      2001      2002      2003
                                               ========  ========  ========  ========  ========  ========  =======
Total Gas Production (BCF)                     298.822   292.561   283.164   282.113   276.153   267.164   265.216
Total Fed. Gas Production (BCF)                298.822   292.561   283.164   282.113   276.153   267.164   265.216
Total Liquids Production (Bil Bbl)               4.614     4.511     4.132     4.092     4.032     3.837     3.779
Average Gas Price ($/MCF)                         1.80      1.56      1.71      1.90      1.94      1.99      2.04
Average Oil Price ($/Bbl)                          .00       .00       .00       .00       .00       .00       .00
Total Gross Revenues (Million $)               536.541   455.802   483.670   535.452   536.720   532.476   540.587
 Total Gravity/Trans. Cost Adj.                   .000      .000      .000      .000      .000      .000      .000
Total Adjusted Revenues                        536.541   455.802   483.670   535.452   536.720   532.476   540.587
 Total Royalties                                67.068    56.975    60.459    66.931    67.090    66.560    67.573
 Total Federal Royalties                          .000      .000      .000      .000      .000      .000      .000
 Total Private Royalties                        67.068    56.975    60.459    66.931    67.090    66.560    67.573
Total Net Sales                                469.473   398.827   423.211   468.520   469.630   465.917   473.014
Total Operating Cost                           142.412   132.919   126.647   124.273   119.368   114.571   108.802
 Total G&A on Expensed Items (on Gen. O&M)      18.059    17.190    16.372    15.749    14.842    13.534    12.792
 Total G&A on Capitalized Items (on tot.inv.    11.501     3.441     3.569     4.021     4.354     6.611     5.960
 Total Pressure Maint./Cycling                    .000      .000      .000      .000      .000      .000      .000
 Total General O&M (Fixed,Var.,Comp.)           72.234    68.493    68.670    69.458    68.944    66.493    66.485
 Total Environmental O&M Costs                    .000      .265    -3.183    -6.463    -9.575   -12.356   -15.316
 Total Stimulation Costs                          .195     3.522      .990     1.617     1.770     2.701     1.887
 Total Recompletion Costs                         .000      .000      .000      .000      .000      .000      .000
Total Intangible Investment                      5.658    27.988    28.762    31.227    33.101    52.047    45.524
 Intang. Exploratory Costs                       4.492     3.077     3.406     2.181     4.473     6.071     5.964
 Intang. Development Costs                       1.166     6.345     4.978     7.271     8.108    10.450    10.773
 Intang. Environmental Costs                      .000    18.566    20.379    21.775    20.520    35.526    28.788
 Other Intang. Costs                              .000      .000      .000      .000      .000      .000      .000
Portion of Intangibles to Capitalize             1.697     2.827     2.515     2.836     3.775     4.956     5.021
TOTAL INVESTMENTS                              115.014    34.413    35.694    40.213    43.538    66.114    59.602
Tangible Investments                           109.356     6.425     6.932     8.986    10.436    14.067    14.078
 Tang. Exploratory Cost                          1.497     1.041     1.140      .726     1.487     2.012     1.969
 Tang. Development Cost                           .782     4.304     3.352     4.857     5.392     6.944     7.143
 Tang. Environmental Cost                         .965      .086      .106      .116      .120      .140      .164
 Other Tang. Capital                           106.112      .994     2.334     3.288     3.438     4.971     4.801
Total Depreciable/Capitalized Investments      111.053     9.252     9.447    11.822    14.211    19.023    19.098
 Adj. for Federal Tax Credits                     .000      .000      .000      .000      .000      .000      .000
Adj. Depreciable/Capitalize Inv                111.053     9.252     9.447    11.822    14.211    19.023    19.098
Depreciation                                    18.574    28.986    22.383    18.899    17.599    19.791    21.959
Depletable G&G/Lease Acq. Costs                  4.133     1.110     4.422     3.783      .605     1.156     1.679
 Depletable Lease Acq. Cost                      4.082     1.075     4.383     3.758      .554     1.087     1.611
 Depletable G&G Costs                             .051      .035      .039      .025      .051      .069      .068
 Adjustments for Federal Tax Credits              .000      .000      .000      .000      .000      .000      .000
Depletion Base                                   4.133     1.110     4.422     3.783      .605     1.156     1.679
Expensed G&G/Lease Costs                          .264      .180      .200      .128      .263      .358      .352
 Expensed Lease Acq. Cost                         .000      .000      .000      .000      .000      .000      .000
 Expensed G&G Costs                               .264      .180      .200      .128      .263      .358      .352
Operating Cost/mcf                                .477      .454      .447      .441      .432      .429      .410

 Industry Jobs (Thou. - Rev Based)                  2.        1.        2.        2.        2.        2.        2.
 Industry Jobs (Thou. - Cost Based)                 2.        2.        2.        2.        2.        2.        2.

 Total Jobs (Thou. - Rev Based)                     5.        4.        4.        5.        5.        5.        5.
 Total Jobs (Thou. - Cost Based)                    7.        4.        4.        4.        4.        5.        4.
```

Table E-10 (continued)

OUTPUT File: RES.OUT

```
Detailed Financial Report
     ========= Resource Aggregations ===========

                                       Year       1997     1998     1999     2000     2001     2002     2003
                                                ======== ======== ======== ======== ======== ======== ========
Total Operating Wells                            4116.    4048.    4026.    4028.    4039.    3887.    3870.
  Exploratory Wells (Incl. Dry Holes)              23.      16.      10.       8.      23.      26.      21.
  Total Primary Development Wells                   5.      14.      30.      34.      36.      55.      54.
   - Successful Development Wells                   4.      12.      27.      31.      32.      50.      48.
   - Dry Development Wells                          0.       1.       3.       3.       4.       6.       5.
  Total Infill Wells                                0.      58.       0.       6.       6.      13.       0.
   - Successful Infill Wells                        0.      47.       0.       5.       5.      10.       0.
   - Dry Infill Wells                               0.      12.       0.       1.       1.       3.       0.
  Total Dry Wells  Drilled (Development)            0.      13.       3.       5.       5.       8.       5.

  Total Wells (Expl. & Dev.)                       28.      88.      39.      49.      65.      94.      75.
Net Revenues (Million $)                       469.473  398.827  423.211  468.520  469.630  465.917  473.014
 Operator Severance Taxes                       27.360   23.130   24.253   26.718   27.079   27.081   27.857
 Operating Costs                               142.412  132.919  126.647  124.273  119.368  114.571  108.802
 Expensed Int.,G&G, and Lease Acq.               4.225   25.342   26.448   28.520   29.590   47.449   40.855
 Depreciation                                   18.574   28.986   22.383   18.899   17.599   19.791   21.959
 Depletion Allowance                              .009     .037     .108     .213     .322     .477     .617
Taxable Income                                 276.894  188.412  223.373  269.897  275.671  256.548  272.924
 Tax Credit Addback                               .000     .000     .000     .000     .000     .000     .000
 Intangible Addback                               .000     .000     .000     .000     .000     .000     .000
 G&G/Lease Addback                                .000     .000     .000     .000     .000     .000     .000
Net Income Before Taxes                        276.894  188.412  223.373  269.897  275.671  256.548  272.924
 State Income Taxes                              5.460    2.782    3.515    4.471    5.040    4.477    5.931
 Federal Income Tax                             92.288   63.115   74.753   90.247   92.017   85.707   90.781
 Federal Tax Credits                              .000     .000     .000     .000     .000     .000     .000
Net Income After Taxes                         179.146  122.515  145.105  175.179  178.614  166.365  176.212
 plus Depreciation                              18.574   28.986   22.383   18.899   17.599   19.791   21.959
 plus Depletion                                   .009     .037     .108     .213     .322     .477     .617
 less Depletable Items                           4.133    1.110    4.422    3.783     .605    1.156    1.679
 less Depreciable/Capitalized Items            111.053    9.252    9.447   11.822   14.211   19.023   19.098
 less Tax Credit on Expensable Items              .000     .000     .000     .000     .000     .000     .000
Annual After Tax Cash Flow                      82.543  141.177  153.727  178.687  181.720  166.453  178.011


Total CO2 Production (BCF)                      22.359   22.237   22.012   21.856   21.543   21.240   21.123
Total Nitrogen Production (BCF)                  6.831    6.786    6.650    6.534    6.374    5.867    5.706
Total Hydro-Sulf Production (k ton)             78.070   78.071   78.071   78.042   78.007   77.980   77.958
```

Table E-11

OUTPUT File: RESERVOIR.OUT
Contains non-associated gas production and related pro-forma entries for each reservoir (GSAMID) in the model if required. The width of the file has been abridged to fit the page; the actual file contains data for the years 1997-2020. A sample reservoir entry is shown below.

```
Detailed Financial Report
     GSAM ID: 01116702F007 Tech.: Current Technology



                                       Year          1997     1998     1999     2000     2001     2002     2003
                                                   ======== ======== ======== ======== ======== ======== ========
Gas Production (BCF)                                 .000     .000     .000     .000     .000     .000     .000
 Gas Price ($/MCF)                                  2.850    2.410    1.970    2.040    2.110    2.170    2.240
Liquids Production (MMBO)                            .000     .000     .000     .000     .000     .000     .000
 Oil Price ($/Bbl)                                   .000     .000     .000     .000     .000     .000     .000
Gross Revenues (MM$)                                 .000     .000     .000     .000     .000     .000     .000
 Gravity/Trans. Cost Adj.                            .000     .000     .000     .000     .000     .000     .000
Adjusted Revenues                                    .000     .000     .000     .000     .000     .000     .000
 Royalties                                           .000     .000     .000     .000     .000     .000     .000
Net Sales                                            .000     .000     .000     .000     .000     .000     .000
Total Operating Cost                                 .000     .000     .000     .000     .000     .000     .000
 G&A on Expensed Items (on Gen. O&M)                 .000     .000     .000     .000     .000     .000     .000
 G&A on Capitalized Items (on tot. inv.)             .000     .000     .000     .000     .000     .000     .000
 Pressure Maint./Cycling                             .000     .000     .000     .000     .000     .000     .000
 General O&M (Fixed,Var.,Comp.)                      .000     .000     .000     .000     .000     .000     .000
 Environmental O&M Costs                             .000     .000     .000     .000     .000     .000     .000
 Stimulation Costs                                   .000     .000     .000     .000     .000     .000     .000
 Recompletion Costs                                  .000     .000     .000     .000     .000     .000     .000
Intangible Investment                                .000     .000     .000     .000     .000     .000     .000
 Intang. Exploratory Costs                           .000     .000     .000     .000     .000     .000     .000
 Intang. Development Costs                           .000     .000     .000     .000     .000     .000     .000
 Intang. Envionmental Costs                          .000     .000     .000     .000     .000     .000     .000
 Other Intangible Costs                              .000     .000     .000     .000     .000     .000     .000
Portion of Intangibles to Capitalize                 .000     .000     .000     .000     .000     .000     .000
TOTAL INVESTMENTS                                    .000     .000     .000     .000     .000     .000     .000
Tangible Investments                                 .000     .000     .000     .000     .000     .000     .000
 Tang. Exploratory Cost                              .000     .000     .000     .000     .000     .000     .000
 Tang. Development Cost                              .000     .000     .000     .000     .000     .000     .000
 Tang. Environmental Cost                            .000     .000     .000     .000     .000     .000     .000
 Other Tang. Cost                                    .000     .000     .000     .000     .000     .000     .000
Total Depreciable/Capitalized Investments            .000     .000     .000     .000     .000     .000     .000
 Adj. for Federal Tax Credits                        .000     .000     .000     .000     .000     .000     .000
Depreciable/Capitalize Base                          .000     .000     .000     .000     .000     .000     .000
Depreciation                                         .000     .000     .000     .000     .000     .000     .000
Depletable G&G/Lease Acq. Costs                      .000     .000     .000     .000     .000     .000     .000
 Depletable Lease Acq. Cost                          .000     .000     .000     .000     .000     .000     .000
 Depletable G&G Costs                                .000     .000     .000     .000     .000     .000     .000
 Adjustments for Federal Tax Credits                 .000     .000     .000     .000     .000     .000     .000
Depletion Base                                       .000     .000     .000     .000     .000     .000     .000
Expensed G&G/Lease Acq. Costs                        .000     .000     .000     .000     .000     .000     .000
 Expensed Lease Acq. Cost                            .000     .000     .000     .000     .000     .000     .000
 Expensed G&G Costs                                  .000     .000     .000     .000     .000     .000     .000
Operating Cost/Mcf                                   .000     .000     .000     .000     .000     .000     .000
```

Table E-11 (continued)

```
Detailed Financial Report
     GSAM ID: 01116702F007 Tech.: Current Technology



                                       Year          1997      1998      1999      2000      2001      2002      2003
                                                   ========  ========  ========  ========  ========  ========  ========
Total Operating Wells                                 .0        .0        .0        .0        .0        .0        .0
  Exploratory Wells                                   .0        .0        .0        .0        .0        .0        .0
  Total Development Wells                             .0        .0        .0        .0        .0        .0        .0
  - Successful Development Wells                      .0        .0        .0        .0        .0        .0        .0
  - Dry Development Wells                             .0        .0        .0        .0        .0        .0        .0
  Total Infill Wells                                  .0        .0        .0        .0        .0        .0        .0
  - Successful Infill Wells                           .0        .0        .0        .0        .0        .0        .0
  - Dry Infill Wells                                  .0        .0        .0        .0        .0        .0        .0
  Total Dry Wells Drilled                             .0        .0        .0        .0        .0        .0        .0
 Total Wells (Expl. & Dev.)                         .000      .000      .000      .000      .000      .000      .000
Net Revenues                                        .000      .000      .000      .000      .000      .000      .000
 Operator Severance Taxes                           .000      .000      .000      .000      .000      .000      .000
 Operating Costs                                    .000      .000      .000      .000      .000      .000      .000
 Expensed Int.,G&G, and Lease Acq.                  .000      .000      .000      .000      .000      .000      .000
 Depreciation                                       .000      .000      .000      .000      .000      .000      .000
 Depletion Allowance                                .000      .000      .000      .000      .000      .000      .000
Taxable Income                                      .000      .000      .000      .000      .000      .000      .000
 Tax Credit Addback                                 .000      .000      .000      .000      .000      .000      .000
 Intangible Addback                                 .000      .000      .000      .000      .000      .000      .000
 G&G/Lease Addback                                  .000      .000      .000      .000      .000      .000      .000
Net Income Before Taxes                             .000      .000      .000      .000      .000      .000      .000
 State Income Taxes                                 .000      .000      .000      .000      .000      .000      .000
 Federal Income Tax                                 .000      .000      .000      .000      .000      .000      .000
 Federal Tax Credits                                .000      .000      .000      .000      .000      .000      .000
Net Income After Taxes                              .000      .000      .000      .000      .000      .000      .000
 plus Depreciation                                  .000      .000      .000      .000      .000      .000      .000
 plus Depletion                                     .000      .000      .000      .000      .000      .000      .000
 less Depletable Items                              .000      .000      .000      .000      .000      .000      .000
 less Depreciable/Capitalized Items                 .000      .000      .000      .000      .000      .000      .000
 less Tax Credit on Expensable Items                .000      .000      .000      .000      .000      .000      .000
Annual After Tax Cash Flow                          .000      .000      .000      .000      .000      .000      .000
Discounted After Tax Cash Flow                      .000      .000      .000      .000      .000      .000      .000
Cumulative Discounted After Tax Cash Flow           .000      .000      .000      .000      .000      .000      .000
```

Table E-12

OUTPUT File: STATE.OUT
Contains non-associated gas production and related pro-forma entries by state. The width of the file has been abridged to fit the page; the actual file contains data for the years 1997-2020. A sample state entry is shown below.

```
Detailed Financial Report
      Case: Current Technology   State Name = ALABAMA



                                  Year          1997      1998      1999      2000      2001      2002      2003
                                             ========  ========  ========  ========  ========  ========  ========
Total Gas Production (BCF)                    329.505   347.815   351.705   359.087   361.165   357.353   350.482
Total Fed. Gas Production (BCF)                80.539    80.539    79.166    67.154    49.489    35.541    26.655
Average Gas Price ($/MCF)                        2.04      1.85      1.65      1.75      1.85      1.94      2.04
Total Liquids Production (Bil Bbl)               .000      .000      .000      .000      .000      .000      .000
Oil Price ($/Bbl)                                 .00       .00       .00       .00       .00       .00       .00
Total Gross Revenues (MM$)                    673.282   642.911   580.762   626.949   666.754   692.311   714.939
 Total Gravity/Trans. Cost Adj.                  .000      .000      .000      .000      .000      .000      .000
Total Adjusted Revenues                       673.282   642.911   580.762   626.949   666.754   692.311   714.939
 Total Royalties                               84.160    80.364    72.595    78.369    83.344    86.539    89.367
 Total Federal Royalties                       20.413    18.666    16.625    14.958    11.653     8.743     6.860
 Total Private Royalties                       63.747    61.698    55.970    63.410    71.692    77.796    82.508
Total Net Sales                               589.122   562.547   508.167   548.580   583.410   605.772   625.572
Total Operating Cost                          135.912   124.210   114.552   121.024   126.691   128.974   131.267
 Total G&A on Expensed Items (on Gen. O&M)     18.464    18.573    17.901    18.376    19.019    19.557    20.148
 Total G&A on Capitalized Items (on tot.inv.   22.526    10.489     5.817     9.232     9.541    10.820    10.297
 Total Pressure Maint./Cycling                   .000      .000      .000      .000      .000      .000      .000
 Total General O&M (Fixed,Var.,Comp.)          73.854    74.291    71.604    73.502    76.075    76.546    77.232
 Total Processing O&M                          18.302    18.295    17.782    17.718    18.082    18.004    17.935
 Total Environmental O&M Costs                   .000      .000      .000      .000      .000     1.680     3.360
 Total Stimulation Costs                        2.766     2.562     1.447     2.196     3.975     2.368     2.295
 Total Recompletion Costs                        .000      .000      .000      .000      .000      .000      .000
Total Intangible Investment                    62.096    56.723    31.264    49.448    50.960    63.432    60.602
 Intang. Exploratory Costs                       .000      .000      .000      .000      .000      .000      .000
 Intang. Development Costs                     62.096    56.723    31.264    49.448    50.960    50.219    47.344
 Env. Intangible Capital Costs                   .000      .000      .000      .000      .000    13.213    13.258
 Other Intangible Costs                          .000      .000      .000      .000      .000      .000      .000
Portion of Intangibles to Capitalize           18.629    17.017     9.379    14.834    15.288    15.066    14.203
TOTAL INVESTMENTS                             225.264   104.886    58.173    92.322    95.412   108.203   102.971
Tangible Investments                          163.168    48.163    26.909    42.874    44.452    44.771    42.369
 Tang. Exploratory Cost                          .000      .000      .000      .000      .000      .000      .000
 Tang. Development Cost                        41.285    38.118    21.232    33.429    34.258    33.594    31.503
 Tang. Environmental Cost                        .400      .028      .034      .024      .021      .025      .025
 Other Tangible Cost                          121.483    10.017     5.644     9.421    10.173    11.153    10.841
Total Depreciable/Capitalized Investments     181.797    65.180    36.289    57.709    59.740    59.837    56.572
 Adj. for Federal Tax Credits                    .000      .000      .000      .000      .000      .000      .000
Adj. Depreciable/Capitalize Inv.              181.797    65.180    36.289    57.709    59.740    59.837    56.572
Depreciation                                   25.870    54.072    52.805    51.127    53.342    61.714    64.354
Depletable G&G/Lease Acq. Costs                77.883      .355      .572      .000      .964     1.388      .000
 Depletable Lease Acq. Cost                    77.883      .355      .572      .000      .964     1.388      .000
 Depletable G&G Costs                            .000      .000      .000      .000      .000      .000      .000
 Adjustments for Federal Tax Credits             .000      .000      .000      .000      .000      .000      .000
Depletion Base                                 77.883      .355      .572      .000      .964     1.388      .000
Expensed G&G/Lease Acq. Costs                    .000      .000      .000      .000      .000      .000      .000
 Expensed Lease Acq. Cost                        .000      .000      .000      .000      .000      .000      .000
 Expensed G&G Costs                              .000      .000      .000      .000      .000      .000      .000
Operating Cost/Mcf                               .412      .357      .326      .337      .351      .361      .375

 Industry Jobs (Thou.- Rev Based)                  2.        2.        2.        2.        2.        2.        2.
 Industry Jobs (Thou. - Cost Based)                3.        2.        2.        2.        2.        2.        2.

 Total Jobs (Thou. - Rev Based)                    6.        6.        5.        6.        6.        6.        6.
 Total Jobs (Thou. - Cost Based)                  10.        6.        5.        6.        6.        6.        6.
```

Table E-12 (continued)

```
Detailed Financial Report
      Case: Current Technology   State Name = ALABAMA


                                   Year        1997      1998      1999      2000      2001      2002      2003
                                             ========  ========  ========  ========  ========  ========  ========
  Exploratory Wells (Incl. Dry Holes)              0.        0.        0.        0.        0.        0.        0.
  Total Primary Development Wells                 39.       38.       24.       37.       39.       44.       43.
   - Successful Development Wells                 32.       30.       20.       29.       31.       35.       35.
   - Dry Development Wells                         8.        7.        5.        7.        8.        9.        9.
  Total Infill Wells                               0.        0.        0.        0.       14.        0.        1.
   - Successful Infill Wells                       0.        0.        0.        0.       11.        0.        1.
   - Dry Infill Wells                              0.        0.        0.        0.        3.        0.        0.
  Total Dry Wells Drilled (Development)            8.        7.        5.        7.       10.        9.        9.

  Total Wells (Expl. & Dev.)                      39.       38.       24.       37.       53.       44.       44.
Net Revenues                                  589.122   562.547   508.167   548.580   583.410   605.772   625.572
 Operator Severance Taxes                      58.912    56.255    50.817    54.858    58.341    60.577    62.557
 Operating Costs                              135.912   124.210   114.552   121.024   126.691   128.974   131.267
 Expensed Int.,G&G, and Lease Acq.             43.467    39.706    21.885    34.613    35.672    48.366    46.399
 Depreciation                                  25.870    54.072    52.805    51.127    53.342    61.714    64.354
 Depletion Allowance                             .466      .887     1.129     1.530     1.886     2.079     2.118
 Industry Jobs
Taxable Income                                324.494   287.417   266.978   285.428   307.477   304.061   318.878
 Tax Credit Addback                              .000      .000      .000      .000      .000      .000      .000
 Intangible Addback                              .000      .000      .000      .000      .000      .000      .000
 G&G/Lease Addback                               .000      .000      .000      .000      .000      .000      .000
Net Income Before Taxes                       324.494   287.417   266.978   285.428   307.477   304.061   318.878
 State Income Taxes                            11.694    10.518    10.106    11.489    13.609    14.320    15.570
 Federal Income Tax                           106.352    94.146    87.337    93.139    99.915    98.512   103.125
 Federal Tax Credits                             .000      .000      .000      .000      .000      .000      .000
Net Income After Taxes                        206.448   182.754   169.536   180.800   193.953   191.229   200.183
 plus Depreciation                             25.870    54.072    52.805    51.127    53.342    61.714    64.354
 plus Depletion                                  .466      .887     1.129     1.530     1.886     2.079     2.118
 less Depletable Items                         77.883      .355      .572      .000      .964     1.388      .000
 less Depreciable/Capitalized Items           181.797    65.180    36.289    57.709    59.740    59.837    56.572
 less Tax Credit on Expensable Items             .000      .000      .000      .000      .000      .000      .000
Annual After Tax Cash Flow                    -26.896   172.178   186.610   175.749   188.477   193.797   210.082


Total Carbon Dioxide Production (BCF)            .000      .000      .000      .000      .000      .000      .000
Total Nitrogen Production (BCF)                 3.341     3.337     3.029     3.030     3.317     3.280     3.243
Total Hydro-Sulfide Production (K-Ton)         66.394    66.547    65.386    65.694    67.148    67.696     68.009
```

Table E-13

OUTPUT File: REMAINING.OUT
Contains remaining resource database with all the pertinent information for each reservoir. The width of the file has been abridged to fit the page.

```
01116701F005     1.700     1.400     1.500  4.6900  4.7000     3.700     3.100     3.200  7.0000  7.0100     1.100      .900     1.000
01116701F006      .000      .000      .000  3.4100  3.4100      .000      .000      .000  4.9500  4.9600      .000      .000      .000
01116701F007      .000      .000      .000  2.4500  2.4600      .000      .000      .000  3.6700  3.6800      .000      .000      .000
01116701F008      .000      .000      .000  1.7600  1.7700      .000      .000      .000  2.6300  2.6400      .000      .000      .000
01116701F009      .000      .000      .000  1.2800  1.2800      .000      .000      .000  1.9300  1.9300      .000      .000      .000
01116701F010      .000      .000      .000   .8300   .8300      .000      .000      .000  1.2300  1.2400      .000      .000      .000
01116701F011      .000      .000      .000   .6000   .6100      .000      .000      .000   .9000   .9200      .000      .000      .000
01116701F012      .000      .000      .000   .4500   .8200      .000      .000      .000   .6800  1.2200      .000      .000      .000
01116701F013      .000      .000      .000   .3400   .3400      .000      .000      .000   .5100   .5100      .000      .000      .000
01116701F014      .000      .000      .000   .2800   .5300      .000      .000      .000   .4000   .7600      .000      .000      .000
01116701F015      .000      .000      .000   .2100   .3300      .000      .000      .000   .2900   .4600      .000      .000      .000
01116701F016      .000      .000      .000   .2000   .3100      .000      .000      .000   .2200   .4200      .000      .000      .000
01116701F017      .000      .000      .000   .2000   .2000      .000      .000      .000   .2000   .2000      .000      .000      .000
01116702F005     1.700     1.400     1.500  4.6900  4.7000     3.700     3.100     3.200  7.0100  7.0200     1.100      .900     1.000
01116702F006      .000      .000      .000  3.4100  3.4200      .000      .000      .000  4.9600  4.9600      .000      .000      .000
01116702F007     7.200     6.100     6.400  2.4600  2.4600    13.500    11.300    12.000  3.6800  3.6900     5.000     4.200     4.400
01116702F008      .000      .000      .000  1.7700  1.7700      .000      .000      .000  2.6400  2.6400      .000      .000      .000
01116702F009      .000      .000      .000  1.2800  1.2800      .000      .000      .000  1.9300  1.9400      .000      .000      .000
01116702F010      .000      .000      .000   .8300   .8300      .000      .000      .000  1.2400  1.2400      .000      .000      .000
01116702F011      .000      .000      .000   .6000   .6100      .000      .000      .000   .9000   .9200      .000      .000      .000
01116702F012      .000      .000      .000   .4500   .8200      .000      .000      .000   .6800  1.2200      .000      .000      .000
01116702F013      .000      .000      .000   .3400   .3400      .000      .000      .000   .5100   .5100      .000      .000      .000
01116702F014      .000      .000      .000   .2800   .5300      .000      .000      .000   .4000   .7600      .000      .000      .000
01116702F015      .000      .000      .000   .2100   .3300      .000      .000      .000   .2900   .4600      .000      .000      .000
01116702F016      .000      .000      .000   .2000   .3100      .000      .000      .000   .2200   .4200      .000      .000      .000
01116702F017      .000      .000      .000   .2000   .2000      .000      .000      .000   .2000   .2000      .000      .000      .000
01116703F005      .000      .000      .000  4.6900  4.7000      .000      .000      .000  7.0100  7.0200      .000      .000      .000
01116703F006     3.500     2.900     3.100  3.4100  3.4200     6.800     5.700     6.000  4.9600  4.9600     2.600     2.100     2.300
01116703F007      .000      .000      .000  2.4600  2.4600      .000      .000      .000  3.6800  3.6900      .000      .000      .000
01116703F008      .000      .000      .000  1.7700  1.7700      .000      .000      .000  2.6400  2.6400      .000      .000      .000
01116703F009      .000      .000      .000  1.2800  1.2800      .000      .000      .000  1.9300  1.9400      .000      .000      .000
01116703F010      .000      .000      .000   .8300   .8300      .000      .000      .000  1.2400  1.2400      .000      .000      .000
01116703F011      .000      .000      .000   .6000   .6100      .000      .000      .000   .9000   .9200      .000      .000      .000
01116703F012      .000      .000      .000   .4500   .8200      .000      .000      .000   .6800  1.2200      .000      .000      .000
01116703F013      .000      .000      .000   .3400   .3400      .000      .000      .000   .5100   .5100      .000      .000      .000
01116703F014      .000      .000      .000   .2800   .5300      .000      .000      .000   .4000   .7600      .000      .000      .000
01116703F015      .000      .000      .000   .2100   .3300      .000      .000      .000   .2900   .4600      .000      .000      .000
01116703F016      .000      .000      .000   .2000   .3100      .000      .000      .000   .2200   .4200      .000      .000      .000
```

.

.

.

(continues with other reservoirs)

Description of File: REMAINING.OUT

Data Element	Description	Format
1	12-digit GSAMID	A12, 1x
2	OGIP with Current Technology, Pay Grade 1 and Primary development type	F9.3, 1x
3	Reserves with Current Technology, Pay Grade 1 and Primary development type	F9.3, 1x
4	Reserves with Current Technology, Pay Grade 1 and Infill development type	F9.3, 1x
5	MASP with Current Technology, Pay Grade 1 and Primary development type	F7.4, 1x
6	MASP with Current Technology, Pay Grade 1 and Infill development type	F7.4, 1x
7	OGIP with Current Technology, Pay Grade 2 and Primary development type	F9.3, 1x
8	Reserves with Current Technology, Pay Grade 2 and Primary development type	F9.3, 1x
9	Reserves with Current Technology, Pay Grade 2 and Infill development type	F9.3, 1x
10	MASP with Current Technology, Pay Grade 2 and Primary development type	F7.4, 1x
11	MASP with Current Technology, Pay Grade 2 and Infill development type	F7.4, 1x
12	OGIP with Current Technology, Pay Grade 3 and Primary development type	F9.3, 1x
13	Reserves with Current Technology, Pay Grade 3 and Primary development type	F9.3, 1x
14	Reserves with Current Technology, Pay Grade 3 and Infill development type	F9.3, 1x
15	MASP with Current Technology, Pay Grade 3 and Primary development type	F7.4, 1x
16	MASP with Current Technology, Pay Grade 3 and Infill development type	F7.4, 1x
17	OGIP with Advanced Technology, Pay Grade 1 and Primary development type	F9.3, 1x
18	Reserves with Advanced Technology, Pay Grade 1 and Primary development type	F9.3, 1x
19	Reserves with Advanced Technology, Pay Grade 1 and Infill development type	F9.3, 1x
20	MASP with Advanced Technology, Pay Grade 1 and Primary development type	F7.4, 1x
21	MASP with Advanced Technology, Pay Grade 1 and Infill development type	F7.4, 1x
22	OGIP with Advanced Technology, Pay Grade 2 and Primary development type	F9.3, 1x
23	Reserves with Advanced Technology, Pay Grade 2 and Primary development type	F9.3, 1x
24	Reserves with Advanced Technology, Pay Grade 2 and Infill development type	F9.3, 1x
25	MASP with Advanced Technology, Pay Grade 2 and Primary development type	F7.4, 1x
26	MASP with Advanced Technology, Pay Grade 2 and Infill development type	F7.4, 1x
27	OGIP with Advanced Technology, Pay Grade 3 and Primary development type	F9.3, 1x
28	Reserves with Advanced Technology, Pay Grade 3 and Primary development type	F9.3, 1x
29	Reserves with Advanced Technology, Pay Grade 3 and Infill development type	F9.3, 1x
30	MASP with Advanced Technology, Pay Grade 3 and Primary development type	F7.4, 1x
31	MASP with Advanced Technology, Pay Grade 3 and Infill development type	F7.4, 1x
32	Additional Data Element (for Refract development type)	Free
33	Additional Data Element (for Refract development type)	Free
34	Additional Data Element (for Refract development type)	Free
35	Additional Data Element (for Refract development type)	Free
36	Additional Data Element (for Refract development type)	Free
37	Additional Data Element (for Refract development type)	Free

Table E-14

OUTPUT File: RESFED.OUT
Contains non-associated gas production and the related pro-forma entries by resource type for federal land reservoirs. The width of the file has been abridged to fit the page; the actual file contains data for the years 1997-2020. Here the section for Conventional Gas is shown only. The rest of the resource sections have identical format.

```
Detailed Financial Report
     ========= Resource Aggregations ===========

 ========== Conventional Gas Resource             =============
                                  Year          1997      1998      1999      2000      2001      2002      2003
                                              ========  ========  ========  ========  ========  ========  ========
Total Gas Production (BCF)                    298.822   292.561   283.164   282.113   276.153   267.164   265.216
Total Fed. Gas Production (BCF)               298.822   292.561   283.164   282.113   276.153   267.164   265.216
Total Liquids Production (Bil Bbl)              4.614     4.511     4.132     4.092     4.032     3.837     3.779
Average Gas Price ($/MCF)                       1.80      1.56      1.71      1.90      1.94      1.99      2.04
Average Oil Price ($/Bbl)                        .00       .00       .00       .00       .00       .00       .00
Total Gross Revenues (Million $)              536.541   455.802   483.670   535.452   536.720   532.476   540.587
 Total Gravity/Trans. Cost Adj.                  .000      .000      .000      .000      .000      .000      .000
Total Adjusted Revenues                       536.541   455.802   483.670   535.452   536.720   532.476   540.587
 Total Royalties                               67.068    56.975    60.459    66.931    67.090    66.560    67.573
 Total Federal Royalties                         .000      .000      .000      .000      .000      .000      .000
 Total Private Royalties                       67.068    56.975    60.459    66.931    67.090    66.560    67.573
Total Net Sales                               469.473   398.827   423.211   468.520   469.630   465.917   473.014
Total Operating Cost                          142.412   132.919   126.647   124.273   119.368   114.571   108.802
 Total G&A on Expensed Items (on Gen. O&M)     18.059    17.190    16.372    15.749    14.842    13.534    12.792
 Total G&A on Capitalized Items (on tot.inv.   11.501     3.441     3.569     4.021     4.354     6.611     5.960
 Total Pressure Maint./Cycling                   .000      .000      .000      .000      .000      .000      .000
 Total General O&M (Fixed,Var.,Comp.)          72.234    68.493    68.670    69.458    68.944    66.493    66.485
 Total Environmental O&M Costs                   .000      .265    -3.183    -6.463    -9.575   -12.356   -15.316
 Total Stimulation Costs                         .195     3.522      .990     1.617     1.770     2.701     1.887
 Total Recompletion Costs                        .000      .000      .000      .000      .000      .000      .000
Total Intangible Investment                     5.658    27.988    28.762    31.227    33.101    52.047    45.524
 Intang. Exploratory Costs                      4.492     3.077     3.406     2.181     4.473     6.071     5.964
 Intang. Development Costs                      1.166     6.345     4.978     7.271     8.108    10.450    10.773
 Intang. Environmental Costs                     .000    18.566    20.379    21.775    20.520    35.526    28.788
 Other Intang. Costs                             .000      .000      .000      .000      .000      .000      .000
Portion of Intangibles to Capitalize            1.697     2.827     2.515     2.836     3.775     4.956     5.021
TOTAL INVESTMENTS                             115.014    34.413    35.694    40.213    43.538    66.114    59.602
Tangible Investments                          109.356     6.425     6.932     8.986    10.436    14.067    14.078
 Tang. Exploratory Cost                         1.497     1.041     1.140      .726     1.487     2.012     1.969
 Tang. Development Cost                          .782     4.304     3.352     4.857     5.392     6.944     7.143
 Tang. Environmental Cost                        .965      .086      .106      .116      .120      .140      .164
 Other Tang. Capital                          106.112      .994     2.334     3.288     3.438     4.971     4.801
Total Depreciable/Capitalized Investments     111.053     9.252     9.447    11.822    14.211    19.023    19.098
 Adj. for Federal Tax Credits                    .000      .000      .000      .000      .000      .000      .000
Adj. Depreciable/Capitalize Inv               111.053     9.252     9.447    11.822    14.211    19.023    19.098
Depreciation                                   18.574    28.986    22.383    18.899    17.599    19.791    21.959
Depletable G&G/Lease Acq. Costs                 4.133     1.110     4.422     3.783      .605     1.156     1.679
 Depletable Lease Acq. Cost                     4.082     1.075     4.383     3.758      .554     1.087     1.611
 Depletable G&G Costs                            .051      .035      .039      .025      .051      .069      .068
 Adjustments for Federal Tax Credits             .000      .000      .000      .000      .000      .000      .000
Depletion Base                                  4.133     1.110     4.422     3.783      .605     1.156     1.679
Expensed G&G/Lease Costs                         .264      .180      .200      .128      .263      .358      .352
 Expensed Lease Acq. Cost                        .000      .000      .000      .000      .000      .000      .000
 Expensed G&G Costs                              .264      .180      .200      .128      .263      .358      .352
Operating Cost/mcf                               .477      .454      .447      .441      .432      .429      .410

 Industry Jobs (Thou. - Rev Based)                 2.        1.        2.        2.        2.        2.        2.
 Industry Jobs (Thou. - Cost Based)                2.        2.        2.        2.        2.        2.        2.

 Total Jobs (Thou. - Rev Based)                    5.        4.        4.        5.        5.        5.        5.
 Total Jobs (Thou. - Cost Based)                   7.        4.        4.        4.        4.        5.        4..
```

Table E-14 (continued)

```
Detailed Financial Report
     ========= Resource Aggregations ===========

                                  Year         1997      1998      1999      2000      2001      2002      2003
                                             ========  ========  ========  ========  ========  ========  ========
Total Operating Wells                           4116.     4048.     4026.     4028.     4039.     3887.     3870.
  Exploratory Wells (Incl. Dry Holes)             23.       16.       10.        8.       23.       26.       21.
  Total Primary Development Wells                  5.       14.       30.       34.       36.       55.       54.
   - Successful Development Wells                  4.       12.       27.       31.       32.       50.       48.
   - Dry Development Wells                         0.        1.        3.        3.        4.        6.        5.
  Total Infill Wells                               0.       58.        0.        6.        6.       13.        0.
   - Successful Infill Wells                       0.       47.        0.        5.        5.       10.        0.
   - Dry Infill Wells                              0.       12.        0.        1.        1.        3.        0.
  Total Dry Wells  Drilled (Development)           0.       13.        3.        5.        5.        8.        5.

  Total Wells (Expl. & Dev.)                      28.       88.       39.       49.       65.       94.       75.
Net Revenues (Million $)                      469.473   398.827   423.211   468.520   469.630   465.917   473.014
 Operator Severance Taxes                      27.360    23.130    24.253    26.718    27.079    27.081    27.857
 Operating Costs                              142.412   132.919   126.647   124.273   119.368   114.571   108.802
 Expensed Int.,G&G, and Lease Acq.              4.225    25.342    26.448    28.520    29.590    47.449    40.855
 Depreciation                                  18.574    28.986    22.383    18.899    17.599    19.791    21.959
 Depletion Allowance                             .009      .037      .108      .213      .322      .477      .617
Taxable Income                                276.894   188.412   223.373   269.897   275.671   256.548   272.924
 Tax Credit Addback                              .000      .000      .000      .000      .000      .000      .000
 Intangible Addback                              .000      .000      .000      .000      .000      .000      .000
 G&G/Lease Addback                               .000      .000      .000      .000      .000      .000      .000
Net Income Before Taxes                       276.894   188.412   223.373   269.897   275.671   256.548   272.924
 State Income Taxes                             5.460     2.782     3.515     4.471     5.040     4.477     5.931
 Federal Income Tax                            92.288    63.115    74.753    90.247    92.017    85.707    90.781
 Federal Tax Credits                             .000      .000      .000      .000      .000      .000      .000
Net Income After Taxes                        179.146   122.515   145.105   175.179   178.614   166.365   176.212
 plus Depreciation                             18.574    28.986    22.383    18.899    17.599    19.791    21.959
 plus Depletion                                  .009      .037      .108      .213      .322      .477      .617
 less Depletable Items                          4.133     1.110     4.422     3.783      .605     1.156     1.679
 less Depreciable/Capitalized Items           111.053     9.252     9.447    11.822    14.211    19.023    19.098
 less Tax Credit on Expensable Items             .000      .000      .000      .000      .000      .000      .000
Annual After Tax Cash Flow                     82.543   141.177   153.727   178.687   181.720   166.453   178.011


Total CO2 Production (BCF)                     22.359    22.237    22.012    21.856    21.543    21.240    21.123
Total Nitrogen Production (BCF)                 6.831     6.786     6.650     6.534     6.374     5.867     5.706
Total Hydro-Sulf Production (k ton)            78.070    78.071    78.071    78.042    78.007    77.980    77.958
```

Table E-15

OUTPUT File: RESALL.OUT
Contains non-associated gas production by resource type and the related pro-forma entries for both private land and federal land reservoirs. The width of the file has been abridged to fit the page; the actual file contains data for the years 1997-2020. Here the section for Conventional Gas is shown only. The rest of the resource sections have identical format.

```
Detailed Financial Report
     ========= Resource Aggregations ===========

 ========== Conventional Gas Resource             =============
                                   Year         1997      1998      1999      2000      2001      2002      2003
                                              ========  ========  ========  ========  ========  ========  ========
Total Gas Production (BCF)                    6945.152  6696.982  6930.796  7160.500  7449.410  7731.452  7893.542
Total Fed. Gas Production (BCF)                298.822   292.561   283.164   282.113   276.153   267.164   265.216
Total Liquids Production (Bil Bbl)             247.488   230.946   222.936   218.297   211.400   203.049   195.437
Average Gas Price ($/MCF)                         2.14      1.86      2.00      2.07      2.15      2.22      2.25
Average Oil Price ($/Bbl)                          .00       .00       .00       .00       .00       .00       .00
Total Gross Revenues (Million $)             14872.000 12431.720 13866.510 14803.370 16028.090 17125.630 17742.750
 Total Gravity/Trans. Cost Adj.                   .000      .000      .000      .000      .000      .000      .000
Total Adjusted Revenues                      14872.000 12431.720 13866.510 14803.370 16028.090 17125.630 17742.750
 Total Royalties                              1859.000  1553.966  1733.313  1850.422  2003.511  2140.704  2217.844
 Total Federal Royalties                          .000      .000      .000      .000      .000      .000      .000
 Total Private Royalties                      1859.000  1553.966  1733.313  1850.422  2003.511  2140.704  2217.844
Total Net Sales                              13013.010 10877.760 12133.190 12952.940 14024.570 14984.910 15524.920
Total Operating Cost                          3001.385  2606.000  2600.109  2499.574  2474.791  2452.360  2385.676
 Total G&A on Expensed Items (on Gen. O&M)     445.639   397.588   394.544   371.481   361.622   345.636   333.626
 Total G&A on Capitalized Items (on tot.inv.   357.803   161.538   190.062   198.828   212.519   251.743   251.300
 Total Pressure Maint./Cycling                    .000      .000      .000      .000      .000      .000      .000
 Total General O&M (Fixed,Var.,Comp.)         1782.557  1587.532  1649.962  1623.147  1647.726  1635.176  1639.400
 Total Environmental O&M Costs                    .000     2.821   -71.783  -137.221  -201.238  -252.630  -304.897
 Total Stimulation Costs                        27.368    79.269    61.626    63.949    74.944    87.069    82.792
 Total Recompletion Costs                         .000      .000      .000      .000      .000      .000      .000
Total Intangible Investment                    602.609  1069.557  1246.374  1287.369  1327.907  1674.796  1630.876
 Intang. Exploratory Costs                     400.675   302.508   358.702   354.989   324.644   279.343   279.020
 Intang. Development Costs                     201.934   395.236   458.317   483.873   579.081   642.585   651.075
 Intang. Environmental Costs                      .000   371.816   429.352   448.503   424.178   752.865   700.779
 Other Intang. Costs                              .000      .000      .000      .000      .000      .000      .000
Portion of Intangibles to Capitalize           180.783   209.323   245.106   251.658   271.118   276.578   279.028
TOTAL INVESTMENTS                             3578.096  1615.378  1900.611  1988.288  2125.179  2517.426  2512.992
Tangible Investments                          2975.461   545.817   654.245   700.916   797.282   842.634   882.132
 Tang. Exploratory Cost                        132.593   101.602   119.580   117.805   107.239    92.024    91.803
 Tang. Development Cost                        133.601   265.154   305.262   321.237   382.771   423.520   428.542
 Tang. Environmental Cost                       19.901      .556      .682      .760      .843      .904      .983
 Other Tang. Capital                          2689.346   178.505   228.722   261.115   306.429   326.185   360.803
Total Depreciable/Capitalized Investments     3156.242   755.140   899.351   952.575  1068.399  1119.212  1161.160
 Adj. for Federal Tax Credits                     .000      .000      .000      .000      .000      .000      .000
Adj. Depreciable/Capitalize Inv               3156.242   755.140   899.351   952.575  1068.399  1119.212  1161.160
Depreciation                                   593.067   879.140   864.908   867.852   905.925  1017.325  1125.698
Depletable G&G/Lease Acq. Costs               1178.147   244.720    67.681    40.511    63.662    18.918    36.739
 Depletable Lease Acq. Cost                   1173.593   241.293    63.611    36.479    59.971    15.740    33.564
 Depletable G&G Costs                            4.554     3.427     4.070     4.032     3.691     3.178     3.175
 Adjustments for Federal Tax Credits              .000      .000      .000      .000      .000      .000      .000
Depletion Base                                1178.147   244.720    67.681    40.511    63.662    18.918    36.739
Expensed G&G/Lease Costs                        23.608    17.769    21.102    20.904    19.136    16.476    16.462
 Expensed Lease Acq. Cost                         .000      .000      .000      .000      .000      .000      .000
 Expensed G&G Costs                             23.608    17.769    21.102    20.904    19.136    16.476    16.462
Operating Cost/mcf                                .432      .389      .375      .349      .332      .317      .302

 Industry Jobs (Thou. - Rev Based)                 48.       40.       44.       47.       51.       55.       57.
 Industry Jobs (Thou. - Cost Based)                63.       40.       43.       43.       44.       47.       47.

 Total Jobs (Thou. - Rev Based)                   134.      112.      125.      133.      144.      154.      160.
 Total Jobs (Thou. - Cost Based)                  176.      113.      120.      120.      123.      133.      131..
```

Table E-15 (continued)

```
Detailed Financial Report
     ========= Resource Aggregations ===========

                                      Year        1997      1998      1999      2000      2001      2002      2003
                                               ========  ========  ========  ========  ========  ========  ========
Total Operating Wells                            89253.    80151.    83496.    80493.    80281.    77548.    76888.
  Exploratory Wells (Incl. Dry Holes)              711.      603.      715.      691.      589.      470.      448.
  Total Primary Development Wells                  678.     1113.     1391.     1561.     1843.     1987.     2265.
   - Successful Development Wells                  610.     1002.     1252.     1405.     1658.     1788.     2039.
   - Dry Development Wells                          68.      111.      139.      156.      184.      199.      227.
  Total Infill Wells                                0.      528.      100.       75.       72.      187.       31.
   - Successful Infill Wells                        0.      422.       80.       60.       58.      150.       25.
   - Dry Infill Wells                               0.      106.       20.       15.       14.       37.        6.
  Total Dry Wells  Drilled (Development)            68.      217.      159.      171.      199.      236.      233.

  Total Wells (Expl. & Dev.)                      1388.     2244.     2206.     2327.     2503.     2644.     2744.
Net Revenues (Million $)                      13013.010 10877.760 12133.190 12952.940 14024.570 14984.910 15524.920
 Operator Severance Taxes                       929.397   784.327   871.724   928.095  1005.839  1075.383  1111.278
 Operating Costs                               3001.385  2606.000  2600.109  2499.574  2474.791  2452.360  2385.676
 Expensed Int.,G&G, and Lease Acq.              445.435   878.005  1022.370  1056.613  1075.924  1414.695  1368.306
 Depreciation                                   593.067   879.140   864.908   867.852   905.925  1017.325  1125.698
 Depletion Allowance                              4.378    11.852    21.164    31.679    43.587    54.127    62.864
Taxable Income                                 8039.284  5718.413  6752.840  7569.048  8518.307  8970.722  9470.796
 Tax Credit Addback                                .000      .000      .000      .000      .000      .000      .000
 Intangible Addback                                .000      .000      .000      .000      .000      .000      .000
 G&G/Lease Addback                                 .000      .000      .000      .000      .000      .000      .000
Net Income Before Taxes                        8039.284  5718.413  6752.840  7569.048  8518.307  8970.722  9470.796
 State Income Taxes                             166.346   113.958   134.322   144.383   158.098   163.816   176.797
 Federal Income Tax                            2676.832  1905.519  2250.293  2524.427  2842.528  2994.459  3160.102
 Federal Tax Credits                               .000      .000      .000      .000      .000      .000      .000
Net Income After Taxes                         5196.168  3698.936  4368.290  4900.402  5517.811  5812.739  6134.286
 plus Depreciation                              593.067   879.140   864.908   867.852   905.925  1017.325  1125.698
 plus Depletion                                   4.378    11.852    21.164    31.679    43.587    54.127    62.864
 less Depletable Items                         1178.147   244.720    67.681    40.511    63.662    18.918    36.739
 less Depreciable/Capitalized Items            3156.242   755.140   899.351   952.575  1068.399  1119.212  1161.160
 less Tax Credit on Expensable Items               .000      .000      .000      .000      .000      .000      .000
Annual After Tax Cash Flow                     1459.232  3590.046  4287.356  4806.818  5335.216  5745.930  6124.722


Total CO2 Production (BCF)                      100.673    94.848    91.758    86.541    86.343    88.007    89.167
Total Nitrogen Production (BCF)                 178.934   168.279   172.664   177.533   177.807   177.591   175.766
Total Hydro-Sulf Production (k ton)             518.231   481.606   425.751   414.708   410.863   391.471   388.241
```

Table E-16

OUTPUT File: REGION.PRD

Contains non-associated gas production by region. for the years 1997-2015. A sample region entry is shown below.

```
Region ::::: Appalachia
  Yearly gas production (BCF)

        1997      640.384100
        1998      517.401000
        1999      565.668400
        2000      611.655200
        2001      657.524800
        2002      707.224300
        2003      765.478500
        2004      828.234400
        2005      874.341400
        2006      907.993200
        2007      942.383100
        2008      965.624100
        2009      990.887500
        2010     1011.909000
        2011     1038.732000
        2012     1059.965000
        2013     1081.657000
        2014     1121.409000
        2015     1163.916000
```

Table E-17

OUTPUT File: STATE.PRD

Contains non-associated gas production by state for the years 1997-2020. A sample state entry is shown below.

```
State ::::: LOUISIANA Fed. & State Offshore
  Yearly gas production (BCF)

        1997      338.994100
        1998      694.145300
        1999     1202.507000
        2000     1688.538000
        2001     1942.073000
        2002     2271.816000
        2003     2448.736000
        2004     2581.759000
        2005     2881.821000
        2006     3144.238000
        2007     3397.868000
        2008     3646.989000
        2009     3850.830000
        2010     4250.923000
        2011     4753.062000
        2012     5024.693000
        2013     4408.918000
        2014     3859.310000
        2015     3518.946000
        2016     3263.892000
        2017     3205.398000
        2018     2929.942000
        2019     2824.072000
        2020     2961.494000
```

APPENDIX F
STORAGE RESERVOIR PERFORMANCE MODULE FILES

Appendix F - Storage Reservoir Performance Module Files

CONTENTS

Table F-1

Input Data File: STODIS.STO (Location: SRPM)
This file contains the SRPM database for existing storage reservoirs (generated primarily from the AGA database).

```
RESERVOIRCOMP                                                                                 feet    feet    feet   psig    ACRE    ACRE   I/O  OBSV. COMP HORSE     mmcf     mmcf               mmcf  mmcf/d perc     md frac frac frac  API   GAS
        IDCODE FIELD NAME                      RESERVOIR NAME               ST STID DISC ACTV   PAY MAXDPTH MINDPTH PRESSR  LIMITS   TOTAL WELLS WELLS STAT POWER TOTLBASE TOTLWORK TOTLFUTR  ULTSTCAP  MAXDEL  POR   PERM  SOI  SGI  SWI GRAV  GRAV
02706719218  56 Zoar                           Onondaga                     NY 9900 1888 1916   30.0  1990.0  1583.0  650.0      .0  5612.0    30     1    1  1200   1650.0    502.0       .0    2250.0   40.00 15.0   59.6  .00  .70  .30   .0  .000
02706720448  11 North Summit                   Huntersville Chert           PA 9900 1937 1991  150.0  7574.0  6394.0 3000.0  2058.0  4809.0    20     8    2  6600  10351.0  11500.0  10187.0   23000.0  300.00  8.0   22.1  .00  .70  .30   .0  .000
02706721201  19 Dundee                         Oriskany                     NY 9900 1930 1940   15.0  2202.0  1671.0  765.0 19638.0 18147.0   115    19    1  2280   7075.0   3785.0       .0   11360.0   80.00 12.0   50.0  .00  .70  .30 42.0  .000
02706721203  19 Greenwood                      Oriskany                     NY 9900 1942 1942   10.0  4839.0  4180.0 1900.0  4306.0  3788.0     5     2    1  1240   3025.0    300.0       .0    3725.0    3.00 12.0   50.0  .00  .70  .30 42.0  .000
02706721204  19 North Greenwood                Oriskany                     NY 9900 1942 1971   10.0  4813.0  4747.0 1900.0  2140.0  2658.0     2     0    1  1240   2100.0    500.0       .0    3200.0    2.00 12.0   50.0  .00  .70  .30 42.0  .000
02706721205  21 Woodhull                       Oriskany                     NY 9900 1938 1957   20.0  3977.0  3975.0 1950.0 10800.0 15900.0    50     1    1 11100  17427.0  18427.0   1485.0   35904.0  357.00 12.0   50.0  .00  .70  .30 42.0  .000
02706721217  56 Tuscarora                      Oriskany                     NY 9900 1944 1950   20.0  3885.0      .0 2040.0      .0  4848.0     7     1    1  1440   2586.0   2658.0       .0    6386.0   60.00 12.0   50.0  .00  .70  .30 42.0  .000
02706721219  69 Beech Hill                     Oriskany                     NY 9900 1938 1980   30.0  4912.0      .0 2000.0  1751.0  4888.0    32     9    1  8350  13100.0   9595.0       .0   17800.0   66.00 12.0   50.0  .00  .70  .30 42.0  .000
02706721220  69 East Independence              Oriskany                     NY 9900 1940 1972   30.0  4937.0      .0 2000.0   650.0  2847.0     7     4    1  5000   4200.0   1560.0       .0    6100.0   20.00 12.0   50.0  .00  .70  .30 42.0  .000
02706721221  69 West Independence              Oriskany                     NY 9900 1939 1980   30.0  4783.0      .0 2000.0  1028.0  4239.0    23     7    1  5000   4500.0   4089.0       .0   11600.0   50.00 12.0   50.0  .00  .70  .30 42.0  .000
02706721257  20 Blackhawk                      Oriskany Ss.                 PA 9900 1936 1969     .0  4673.0      .0 2000.0  2400.0  3364.0     3     5    1   900   1700.0    955.0       .0    2500.0   10.00 12.0   50.0  .00  .70  .30 42.0  .000
02706721258  19 Artemas "A"                    Oriskany                     PA 9900 1962 1972   50.0  6095.0  5643.0 2075.0 10476.0 10216.0    15     1    1  4960   8107.0   5850.0       .0   13957.0  153.80 12.0   50.0  .00  .70  .30 42.0  .000
02706721259  19 Artemas "B"                    Oriskany                     PA 9900 1963 1972   50.0  4931.0  4774.0 1815.0  8596.0  7753.0     3     0    1  4960   1247.0    900.0       .0    2147.0   16.90 12.0   50.0  .00  .70  .30 42.0  .000
02706721264  11 Ellisburg                      Oriskany                     PA 9900 1933 1963   15.0  5154.0      .0 2125.0  8050.0 12740.0    84     3    2 24690  45900.0  52530.0  14053.0   98430.0 1046.00 18.0   50.0  .00  .70  .30 42.0  .000
02706721265  11 Greenlick                      Oriskany                     PA 9900 1955 1961   15.0  6674.0  6306.0 4240.0 16547.0 20160.0    46    10    1 13600  27030.0  28830.0   7546.0   55860.0  912.00 12.0   50.0  .00  .70  .30 42.0  .000
02706721266  11 Harrison                       Oriskany                     PA 9900 1934 1953   32.0  4989.0      .0 2140.0  7250.0 12740.0    45     2    1 11100  13382.0  20178.0   7631.0   34100.0  455.00 12.0   50.0  .00  .70  .30 42.0  .000
02706721267  11 Leidy                          Oriskany                     PA 9900 1950 1959   15.0  6249.0  6249.0 4200.0 17570.0 44407.0    64    51    1 23300  46922.0  55081.0   6732.0  113223.0 1224.00 12.0   50.0  .00  .70  .30 42.0  .000
02706721270  11 Tioga                          Oriskany                     PA 9900 1931 1951   24.0  4412.0  3592.0 1680.0  6800.0 15051.0    28     7    2 10395  12000.0  24000.0   2646.0   36000.0  504.00 16.0   50.0  .00  .70  .30 42.0  .000
02706721271  11 Sabinsville                    Oriskany                     PA 9900 1935 1951   22.0  4951.0  4184.0 2130.0  6800.0 16830.0    40     3    1 10500  17921.0  17697.0   3265.0   35618.0  418.00 15.0   50.0  .00  .70  .30 42.0  .000
02706721272  11 Sharon                         Oriskany                     PA 9900 1938 1948   20.0  4827.0      .0 1900.0  1333.0  3149.0    12     2    0     0   2230.0   2300.0   3400.0    5630.0   25.00 12.0   50.0  .00  .70  .30 42.0  .000
02706721287  56 Hebron                         Oriskany                     PA 9900 1931 1953   18.0  5160.0      .0 2200.0      .0  8660.0    49     4    1 16200  12010.0  12062.0       .0   30355.0  180.00 12.0   50.0  .00  .70  .30 42.0  .000
02706721296  56 Summit                         Oriskany                     PA 9900 1946 1959   20.0  2365.0      .0  810.0      .0  4701.0    44     6    1   880   2600.0    815.0       .0    4200.0   75.00 13.0   50.0  .00  .70  .30 42.0  .000
02706721304  88 Rager Mtn.                     Chert-Oriskany               PA 9900 1965 1971   42.0  8045.0  7680.0 3200.0  5000.0  9214.0     7     2    1  6000  11193.0   9300.0   2314.0   20493.0  120.00 12.0   50.0  .00  .70  .30 42.0  .000
02706725268  11 Oakford                        Murrysville, Fifth           PA 9900 1887 1951  100.0  2252.0  1400.0  600.0  3302.0 33002.0   231    46   14 51250  51001.0  61202.0 103023.0  225426.0  775.00 15.0  200.0  .00  .70  .30 40.0  .000
02706725275  27 Hunters Cave                   Big Injun                    PA 9900 1916 1943    9.0  2315.0  1961.0 1290.0  1902.0  4440.0    22     6    1   750   3252.0   1642.0       .0    4711.0   35.00 12.0  100.0  .00  .70  .30 40.0  .000
02706725300  88 Colvin                         Murrysville                  PA 9900 1924 1943   35.0  2400.0  2300.0  800.0   530.0  1527.0     7     2    1   600   1883.0    510.0       .0    2393.0   86.70 12.0   12.0  .00  .70  .30 40.0  .000
02706725306  88 Webster                        Murrysville                  PA 9900 1923 1945   30.0  2340.0  2158.0  600.0   600.0  1796.0     5     0    1   600    621.0    551.0       .0    1172.0   15.00 12.0   12.0  .00  .70  .30 40.0  .000
02706725356  19 Majorsville Shallow            Salt Sands                   PA 9900 1900 1937   25.0  1890.0  1055.0  771.0 10661.0 10941.0    45     6    1  5280   1583.0    852.0       .0    1900.0    1.00 12.0    8.0  .00  .70  .30 40.0  .000
02706732208  56 Bennington                     Medina                       NY 9900 1919 1951   10.0  1840.0      .0  740.0  5917.0 12575.0    63     1    1   600   3330.0    863.0       .0    5130.0   75.00 10.0   25.0  .00  .70  .30 41.0  .000
02706732209  56 Colden                         Medina                       NY 9900 1926 1952   19.0  2420.0      .0  800.0 10781.0 27549.0   159     7    1  8400  11170.0   7500.0       .0   18720.0  100.00 10.0   25.0  .00  .70  .30 41.0  .000
02706732210  56 Collins                        Medina                       NY 9900 1912 1948   11.0  3260.0  2765.0  970.0  4097.0  8704.0    45     2    1  1200   3830.0   1081.0       .0    6680.0   30.00 10.0   25.0  .00  .70  .30 41.0  .000
02706732211  56 Derby                          Medina                       NY 9900 1946 1950   33.0  1505.0      .0  740.0   654.0  3042.0    12     2    1    75    220.0    134.0       .0     470.0    5.00 10.0   25.0  .00  .70  .30 41.0  .000
02706732212  56 Holland                        Medina                       NY 9900 1925 1949   15.0  2488.0      .0  825.0      .0  4758.0    24     1    1   600   1770.0    475.0       .0    2670.0   25.00 10.0   25.0  .00  .70  .30 41.0  .000
02706732213  56 Lawtons                        Medina                       NY 9900 1917 1946   17.0  2538.0  2362.0  750.0      .0  8792.0    29     2    1  1200   1880.0    512.0       .0    2850.0   21.00 10.0   25.0  .00  .70  .30 41.0  .000
02706732214  56 Nashville                      Medina                       NY 9900 1927 1956   18.0  2930.0      .0  940.0      .0 10540.0    70     1    1  3545   5050.0   2131.0       .0    8980.0   70.00 10.0   25.0  .00  .70  .30 41.0  .000
02706732215  56 Perrysburg                     Medina                       NY 9900 1925 1961   18.0  2835.0      .0  975.0      .0  8109.0    39     1    1  3545   3200.0    932.0       .0    5050.0   50.00 10.0   25.0  .00  .70  .30 41.0  .000
02706732216  56 Sheridan                       Medina                       NY 9900 1910 1937   18.0  2085.0      .0  850.0      .0  6996.0    25     1    1  3545   3310.0    605.0       .0    4410.0   23.00 10.0   25.0  .00  .70  .30 41.0  .000
02706732282  56 Corry                          Medina                       PA 9900 1947 1955   12.0  4485.0      .0 1200.0      .0  2150.0    12     1    1   750   1050.0    194.0       .0    1250.0   30.00 10.0   25.0  .00  .70  .30 41.0  .000
02706737260  19 Donegal                        Gordon St                    PA 9900 1907 1940   10.0  2932.0  2329.0 1250.0 18357.0 18584.0   112     4    1  2480   6318.0   3582.0       .0    9900.0  231.00 22.0   55.0  .00  .70  .30   .0  .000
02706737261  19 Heard                          Big Injun 50ft               PA 9900 1907 1943   10.0  3129.0  1615.0  900.0 11581.0 11696.0    38    13    1  5280   1373.0   1038.0       .0    7527.0    2.00 12.0   10.0  .00  .70  .30   .0  .000
02706737262  19 Holbrook                       Gantz                        PA 9900 1925 1950   25.0  3577.0  2740.0  760.0  5477.0  5767.0    13     2    1   130   1140.0    400.0       .0    1540.0    5.00 12.0   10.0  .00  .70  .30   .0  .000
02706737269  11 South Bend                     100 Ft.                      PA 9900 1902 1951   25.0  1493.0      .0  650.0  4560.0  9450.0    61     4    1  6000  11530.0   5810.0   2401.0   17340.0  200.00 14.0   10.0  .00  .70  .30   .0  .000
02706737274  27 Finleyville                    Fifth                        PA 9900 1927 1934    7.0  2668.0  2590.0     .0   231.0   707.0     4     0    0  3300    331.0    285.0       .0     578.0   42.00 14.0   10.0  .00  .70  .30   .0  .000
02706737276  27 Pratt                          Fifth/50 Ft.                 PA 9900 1899 1947    6.0  3108.0  2515.0     .0  4141.0 10563.0    39     9    1  4400   4717.0   2254.0       .0    7716.0   40.00 14.0   10.0  .00  .70  .30   .0  .000
02706737277  27 Swarts                         Fifty Ft.                    PA 9900 1921 1949   12.0  2874.0  2550.0     .0   478.0  1228.0     6     1    0   750    492.0    470.0       .0    1154.0   28.00 14.0   10.0  .00  .70  .30   .0  .000
02706737280  56 Belmouth                       Cooper                       PA 9900 1914 1938    9.0  1775.0      .0  700.0      .0  3847.0    21     1    1   615    600.0    414.0       .0    1400.0   10.00 14.0   10.0  .00  .70  .30   .0  .000
02706737281  56 Boone Mountain                 5th Venango                  PA 9900 1901 1947   38.0  1408.0   982.0  500.0      .0  3887.0    24     7    1   300   1129.0    560.0       .0    2059.0    9.45 14.0   10.0  .00  .70  .30   .0  .000
02706737283  56 Deerlick                       Tiona                        PA 9900 1888 1950   15.0  1270.0      .0  400.0      .0  3759.0    12     9    1  3960     20.0      5.0       .0     120.0    1.00 14.0   10.0  .00  .70  .30   .0  .000
02706737285  56 East Branch                    Cooper                       PA 9900 1888 1948   40.0  1660.0      .0  875.0      .0 13020.0    90    40    1  3960   9310.0   2331.0       .0   13810.0   60.00 14.0   10.0  .00  .70  .30   .0  .000
02706737286  56 Galbraith                      1st Sheffield                PA 9900 1917 1937   24.0  2675.0  2980.0  910.0      .0  5777.0    26     7    1   900   1048.0    632.0       .0    1948.0   20.00 20.0   25.0  .00  .70  .30   .0  .000
02706737289  56 Keelor                         Cooper                       PA 9900 1908 1946   30.0  1815.0      .0  805.0      .0  3624.0    18    14    1  3960   1550.0   1317.0       .0    2850.0   60.00 14.0   10.0  .00  .70  .30   .0  .000
02706737290  56 Markle                         5th Venango                  PA 9900 1921 1938   17.0  1670.0  1496.0  625.0   336.0  2078.0     4     2    1   900    180.0     40.0       .0     265.0   15.00 14.0   40.0  .00  .70  .30   .0  .000
02706737291  56 Queen                          Queen Sand                   PA 9900 1889 1920   71.0  1090.0      .0  300.0      .0  2754.0    25    13    1   450    645.0    193.0       .0     945.0    3.00 14.0   10.0  .00  .70  .30   .0  .000
02706737292  56 St. Mary's                     5th Venango                  PA 9900 1916 1936   54.0  1180.0      .0  590.0      .0  2138.0     8     5    0     0    311.0    102.0       .0     531.0    2.00 14.0   10.0  .00  .70  .30   .0  .000
02706737295  56 Owl's Nest                     Tiona                        PA 9900 1908 1949   20.0  2067.0  1743.0  825.0      .0  6105.0    24    14    1   150   2110.0    491.0       .0    2760.0   10.00 14.0   10.0  .00  .70  .30   .0  .000
02706737297  56 Swede Hill                     Tiona                        PA 9900 1903 1950   15.0  1705.0      .0  750.0      .0  2757.0    20     5    1  3960    800.0    104.0       .0    1100.0   10.00 14.0   10.0  .00  .70  .30   .0  .000
02706737301  88 Gamble-Hayden                  Bayard                       PA 9900 1929 1943   19.0  2900.0  2775.0 1000.0    60.0  1776.0     7     3    1   600   1727.0   1122.0      6.0    2849.0   20.00 14.0   10.0  .00  .70  .30   .0  .000
02706737302  88 Murrysville                    100 Ft.                      PA 9900 1916 1939   30.0  2000.0  1875.0 1000.0   350.0  1916.0     9     4    1  1600   1716.0   1530.0       .0    3246.0  112.20 14.0   10.0  .00  .70  .30   .0  .000
02706737305  88 Truittsburg                    Bayard                       PA 9900 1925 1949   37.0  1590.0  1474.0  900.0   750.0  2660.0    11     9    1  1320   1549.0   2142.0     49.0    3691.0  102.00 14.0   10.0  .00  .70  .30   .0  .000
02706737355  19 Majorsville Deep               Big Injun;Ninevah;Gordon     PA 9900 1900 1937   20.0  3057.0  1460.0  911.0 16351.0 16972.0   121    15    1  5280   5412.0   3846.0       .0   19111.0   47.00 14.0   10.0  .00  .70  .30   .0  .000
03706720138  82 Accident                       Oriskany Sand                MD 9900 1953 1966  100.0  7900.0  7300.0 3050.0 24524.0 34000.0    75     8    2 11000  46677.0  15301.0   2770.0   64770.0  306.00 12.0   50.0  .00  .70  .30   .0  .000
03706721345  19 Coco "A"                       Oriskany                     WV   47 1944 1950   20.0  6761.0  4955.0 1900.0 16447.0 16447.0    68     0    1  9500  28770.0  15730.0       .0   44500.0  297.40 12.0   50.0  .00  .70  .30 42.0  .000
03706721346  19 Coco "B"                       Oriskany                     WV   47 1944 1951   20.0  5446.0  5041.0 1850.0  5227.0  5299.0    21     1    1  9500   6200.0   3500.0       .0    9700.0  179.00 12.0   50.0  .00  .70  .30 42.0  .000
03706721347  19 Coco "C"                       Oriskany                     WV   47 1944 1957   20.0  5556.0  5094.0 1670.0  8769.0  8841.0    24     0    1  9500   9870.0   7400.0       .0   17270.0  142.00 12.0   50.0  .00  .70  .30 42.0  .000
03706721349  19 Glady                          Chert; Oriskany              WV   47 1954 1964  150.0  6930.0  4914.0 2050.0 69611.0 67980.0    47     8    1  3960  20500.0   9500.0       .0   30000.0  322.00 12.0   50.0  .00  .70  .30 42.0  .000
03706721352  19 Hunt                           Oriskany                     WV   47 1947 1951   20.0  5386.0  5027.0 1325.0  4465.0  4314.0    19     4    1  1100   4680.0    650.0       .0    6080.0   13.00 12.0   50.0  .00  .70  .30 42.0  .000
03706721357  19 Ripley                         Oriskany                     WV   47 1945 1954   20.0  5184.0  4777.0 1835.0 23624.0 23397.0    43     8    1  4500  14050.0   9350.0       .0   23400.0  124.00 12.0   50.0  .00  .70  .30 42.0  .000
```

.

.

.

(continues with other reservoirs)

Description of File: STODIS.STO

Data Element	Description	Format
1	11-character reservoir ID (Storage ID)	1x,a11
2	3-digit company code	1x,i3
3	field name	1x,a30
4	reservoir name	1x,a33
5	2-digit state code	1x,a2
6	4-digit state ID	1x,i4
7	discovery year (year)	1x,i4
8	year activated for storage (year)	1x,i4
9	pay thickness (feet)	1x,f6.1
10	maximum depth of the reservoir (feet)	1x,f7.1
11	minimum depth of the reservoir (feet)	1x,f7.1
12	original pressure (psig)	1x,f6.1
13	approximate acreage reservoir limit (acre)	1x,f7.1
14	approximate acreage total (acre)	1x,f7.1
15	number of output and/or input wells	1x,i5
16	number of pressure control and/or observation wells	1x,i5
17	number of compressors	1x,i4
18	horsepower of compressors (hp)	1x,i5
19	base gas total volume (MMCF)	1x,f8.1
20	working gas total volume (MMCF)	1x,f8.1
21	total future undeveloped/unused capacity (MMCF)	1x,f8.1
22	ultimate storage capacity (MMCF)	1x,f8.1
23	designed maximum deliverability (MMCF/D)	1x,f7.2
24	porosity (%)	1x,f4.1
25	permeability (md)	1x,f6.1
26	oil saturation (fraction)	1x,f4.2
27	gas saturation (fraction)	1x,f4.2
28	water saturation (fraction)	1x,f4.2
29	gas api gravity (°API)	1x,f4.1
30	gas specific gravity (dimensionless)	1x,f5.3

Description of the 11-digit Storage Reservoir ID:

Digit	Description
1-2	Storage Region (Same as GSAM Demand Region, see table D-12 in the appendix section for the Demand and Integrating Modules)
3	Status - For Storage Designates Storage Reservoir Type as follows: **7**:Depleted Storage Gas Reservoir (used in SRPM) **8**:Water Aquifer Storage (used in SRPM) **9**:Salt Cavern Storage (used in SRPM)
4	module as follows: **0**:Existing Gas storage (used in SRPM) **1**:Salt Cavern Storage (used in SRPM)
5-8	USGS play code
9-11	reservoir AGA id if available (or just a counter)

Table F-2

Input Data File: STOUND.STO (Location: \SRPM)
This file contains the SRPM database for potential storage reservoirs (developed from NRG data).

RESERVOIR	COMP							feet	feet	feet	psig	ACRE	ACRE	I/O	OBSV.	COMP	HORSE	mmcf	mmcf	mmcf	mmcf	mmcf/d	perc	md	frac	frac	frac	API	GAS
ID	CODE	FIELD NAME	RESERVOIR NAME	ST	STID	DISC	ACTV	PAY	MAXDPTH	MINDPTH	PRESSR	LIMITS	TOTAL	WELLS	WELLS	STAT	POWER	TOTLBASE	TOTLWORK	TOTLFUTR	ULTSTCAP	MAXDEL	POR	PERM	SOI	SGI	SWI	GRAV	GRAV
11710901030	0	COMPTON LANDING	73095503001 COLUSA	CA	410	1955	2000	160.0	2190.0	2190.0	951.0	.0	440.0	2	0	0	0	.0	.0	511.0	10530.0	.00	28.0	750.0	.00	.75	.25	.0	.000
11710903036	0	KIRBY HILL	73094503001 SOLANO	CA	410	1945	2000	190.0	2850.0	2850.0	2205.0	.0	1060.0	2	0	0	0	.0	.0	118.0	39881.0	.00	20.0	371.0	.00	.67	.33	.0	.000
11710903038	0	MCMULLIN RANCH	73096011001 SAN JOAQUIN	CA	410	1960	2000	25.0	4525.0	4525.0	2658.0	.0	3030.0	8	0	0	0	.0	.0	120.0	42810.0	.00	28.0	597.0	.00	.67	.33	.0	.000
11710903031	0	DUNNIGAN HILLS	73094601001 YOLO	CA	410	1946	2000	135.0	2410.0	2410.0	1060.0	.0	1300.0	0	0	0	0	.0	.0	.0	10737.0	.00	34.0	371.0	.00	.70	.30	.0	.000
11710903044	0	RIVER ISLAND	73095001004 SACRAMENTO	CA	410	1950	2000	40.0	3730.0	3730.0	1810.0	.0	4910.0	21	0	0	0	.0	.0	.0	19116.0	.00	28.0	371.0	.00	.65	.35	.0	.000
11711001052	0	TRICO	74593403002 TULARE	CA	410	1934	2000	24.0	3225.0	3225.0	1480.0	.0	11125.0	12	0	0	0	.0	.0	110.0	84270.0	.00	28.0	1200.0	.00	.73	.27	.0	.000
11711401061	0	SALT LAKE	76090301001 LOS ANGELES	CA	410	1903	2000	925.0	2300.0	2300.0	880.0	.0	1165.0	27	0	0	0	.0	.0	927.0	205500.0	.00	39.0	300.0	.00	.69	.31	.0	.000
11711404056	0	COYOTE, EAST	76091101001 ORANGE	CA	410	1911	2000	800.0	2500.0	2500.0	.0	.0	1505.0	91	0	0	0	.0	.0	829.0	40230.0	.00	26.0	600.8	.00	.68	.32	.0	.000
11711403060	0	INGLEWOOD	76092401002 LOS ANGELES	CA	410	1924	2000	1025.0	2400.0	2400.0	1695.0	.0	1215.0	251	0	0	0	.0	.0	4142.0	107100.0	.00	27.0	281.3	.00	.69	.31	.0	.000
06716308133	0	EATON RAPIDS 36-02N-03W	30597119001 INGHAM	MI	21	1971	2000	55.0	3740.0	3740.0	2030.0	.0	1120.0	0	0	0	0	.0	.0	.0	14691.0	.00	5.1	13.0	.00	.70	.30	.0	.000
06716307141	0	MANISTEE 27-22N-16W	30597460001 MANISTEE	MI	21	1974	2000	224.0	4288.0	4288.0	2223.0	.0	320.0	2	0	0	0	.0	.0	176.0	11910.0	.00	6.2	19.3	.00	.70	.30	.0	.000
06716301123	0	BEAVER CREEK	30594701001 CRAWFORD	MI	21	1947	2000	20.0	4160.0	4160.0	.0	.0	4840.0	0	0	0	0	.0	.0	326.0	21000.0	.00	15.0	6.2	.00	.70	.30	.0	.000
06716308127	0	CAPAC	30596104001 ST CLAIR	MI	21	1961	2000	8.0	4505.0	4505.0	.0	.0	12560.0	0	0	0	0	.0	.0	.0	21365.0	.00	8.7	3.6	.00	.70	.30	.0	.000
06716307128	0	CHESTER 18-30N-02W	30597109001 OTSEGO	MI	21	1971	2000	80.0	5800.0	5800.0	3093.0	.0	840.0	21	0	0	0	.0	.0	721.0	11700.0	.00	8.0	13.0	.00	.70	.30	.0	.000
06716312134	0	GOODWELL	30594309001 NEWAYGO	MI	21	1943	2000	36.0	7721.0	7721.0	3420.0	.0	1240.0	0	0	0	0	.0	.0	171.0	12000.0	.00	19.0	240.0	.00	.95	.05	.0	.000
06716307129	0	COLD SPRINGS 12-28N-06W	30597323001 KALKASKA	MI	21	1973	2000	296.0	6564.0	6564.0	2650.0	.0	400.0	0	0	0	0	.0	.0	.0	17683.0	.00	7.3	19.3	.00	.70	.30	.0	.000
06716307139	0	MANISTEE 23-22N-16W	30597457001 MANISTEE	MI	21	1974	2000	277.0	4220.0	4220.0	2250.0	.0	480.0	4	0	0	0	.0	.0	491.0	21000.0	.00	7.0	19.3	.00	.70	.30	.0	.000
06716307136	0	KALKASKA 21-27N-08W	30597125001 KALKASKA	MI	21	1971	2000	108.0	6562.0	6562.0	3525.0	.0	1040.0	0	0	0	0	.0	.0	231.0	13200.0	.00	9.2	19.3	.00	.70	.30	.0	.000
06716311115	0	ALBION-PULASKI-SCIPIO TRD	30595701001 HILLSDALE	MI	21	1957	2000	50.0	3563.0	3563.0	1601.0	.0	13260.0	0	0	0	0	.0	.0	1042.0	211200.0	.00	4.8	84.5	.00	.70	.30	.0	.000
05716502155	0	ABERDEEN, EAST	20098901001 MONROE	MS	2310	1989	2000	57.0	3076.0	3076.0	1413.0	.0	1920.0	7	0	0	0	.0	.0	5038.0	12750.0	.00	14.0	75.3	.00	.82	.18	.0	.000
05716502015	0	STAR	20097411002 LAMAR	AL	110	1974	2000	34.0	4602.0	4602.0	2034.0	.0	4000.0	17	0	0	0	.0	.0	5098.0	15800.0	.00	.0	75.3	.00	.65	.35	.0	.000
05716503012	0	MT. ZION	20098307001 LAMAR	AL	110	1983	2000	30.0	3672.0	3672.0	1499.0	.0	6400.0	23	0	0	0	.0	.0	2509.0	12120.0	.00	9.0	100.0	.00	.65	.35	.0	.000
05716502016	0	YELLOW CREEK	20098313001 LAMAR	AL	110	1983	2000	74.0	4287.0	4287.0	1952.0	.0	5000.0	11	0	0	0	.0	.0	4133.0	13200.0	.00	10.0	75.3	.00	.65	.35	.0	.000
05716502172	0	MCKINLEY CREEK	20097305001 MONROE	MS	2310	1973	2000	65.0	4363.0	4363.0	2005.0	.0	2720.0	11	0	0	0	.0	.0	2455.0	14280.0	.00	17.0	75.3	.00	.81	.19	.0	.000
05716503162	0	CORINNE	20097201002 MONROE	MS	2310	1972	2000	50.0	2700.0	2700.0	1235.0	.0	8480.0	103	0	0	0	.0	.0	3242.0	48900.0	.00	11.5	75.3	.00	.84	.16	.0	.000
05714946158	0	BAXTERVILLE	21094401003 LAMAR	MS	2310	1944	2000	25.0	4900.0	4900.0	2346.0	.0	7680.0	343	0	0	0	.0	.0	3.0	21780.0	.00	31.0	600.0	.00	.70	.30	.0	.000
05714943182	0	TINSLEY	21093901001 YAZOO	MS	2310	1939	2000	40.0	4366.0	4366.0	2021.0	.0	10160.0	175	0	0	0	.0	.0	3808.0	17400.0	.00	21.0	100.0	.00	.78	.22	.0	.000
05714940168	0	HEIDELBERG AREA	21094407001 JASPER	MS	2310	1944	2000	42.0	4350.0	4350.0	1846.0	.0	8360.0	327	0	0	0	.0	.0	2266.0	17100.0	.00	27.0	300.0	.00	.70	.30	.0	.000
05714937157	0	BAXTERVILLE	21094401001 LAMAR	MS	2310	1944	2000	57.0	8507.0	8507.0	4022.0	.0	7680.0	343	0	0	0	.0	.0	6972.0	85800.0	.00	24.3	1474.0	.00	.85	.15	.0	.000
04714910070	0	BLACKJACK CREEK	21097205001 SANTA ROSA	FL	910	1972	2000	40.0	15790.0	15790.0	7930.0	.0	3360.0	10	0	0	0	.0	.0	4628.0	48750.0	.00	15.8	112.0	.00	.87	.13	.0	.000
09712102018	0	BOUNDARY BUTTE, EAST	59095401001 APACHE	AZ	2	1954	2000	168.0	4400.0	4400.0	1326.0	.0	2800.0	4	0	0	0	.0	.0	256.0	9960.0	.00	8.0	2.3	.00	.70	.30	.0	.000
09714406227	0	JAL, WEST	43096325001 LEA	NM	3010	1963	2000	51.0	11510.0	11510.0	7634.0	.0	640.0	1	0	0	0	.0	.0	164.0	13842.0	.00	8.0	27.0	.00	.68	.32	.0	.000
09719357230	0	TOBAC	43096457001 CHAVES	NM	3010	1964	2000	10.0	9058.0	9058.0	3083.0	.0	4520.0	11	0	0	0	.0	.0	538.0	12690.0	.00	7.0	100.0	.00	.70	.30	.0	.000
09712207208	0	CHA CHA	58095901001 SAN JUAN	NM	3010	1959	2000	20.0	5300.0	5300.0	1630.0	.0	7840.0	29	0	0	0	.0	.0	295.0	19800.0	.00	13.5	57.0	.00	.67	.33	.0	.000
09719357218	0	BOUGH	43094903001 LEA	NM	3010	1949	2000	44.0	9560.0	9560.0	3588.0	.0	2480.0	3	0	0	0	.0	.0	20.0	14144.0	.00	13.0	76.0	.00	.85	.15	.0	.000
09714410224	0	HOBBS	43092803003 LEA	NM	3010	1928	2000	20.0	3150.0	3150.0	1338.0	.0	13840.0	347	0	0	0	.0	.0	158.0	6330.0	.00	11.0	36.5	.00	.65	.35	.0	.000
09712201212	0	TOCITO DOME	58096303001 SAN JUAN	NM	3010	1963	2000	17.0	6338.0	6338.0	3217.0	.0	6380.0	17	0	0	0	.0	.0	1585.0	29250.0	.00	8.6	94.0	.00	.75	.25	.0	.000
09714411219	0	CATO	43096609001 CHAVES	NM	3010	1966	2000	40.0	3496.0	3496.0	1153.0	.0	9320.0	66	0	0	0	.0	.0	1595.0	31110.0	.00	7.5	32.0	.00	.68	.32	.0	.000
09719357216	0	ANDERSON RANCH & NORTH	43095307001 LEA	NM	3010	1953	2000	88.0	9660.0	9660.0	3569.0	.0	2480.0	26	0	0	0	.0	.0	661.0	24966.0	.00	10.0	80.0	.00	.80	.20	.0	.000
09719357215	0	ALLISON	43095403001 ROOSEVELT	NM	3010	1954	2000	94.0	9490.0	9490.0	3460.0	.0	12175.0	27	0	0	0	.0	.0	646.0	51120.0	.00	7.8	200.0	.00	.67	.33	.0	.000
10713005195	0	STENSVAD	51095803001 ROSEBUD	MT	25	1958	2000	68.0	5308.0	5308.0	2082.0	.0	1160.0	7	0	0	0	.0	.0	.0	5132.0	.00	14.2	130.0	.00	.80	.20	.0	.000
10713307514	0	MILL-GILLETTE	51596206001 CAMPBELL	WY	49	1962	2000	30.0	7847.0	7847.0	2767.0	.0	3280.0	13	0	0	0	.0	.0	85.0	9630.0	.00	12.8	123.0	.00	.61	.39	.0	.000
10713307510	0	GAS DRAW	51596807001 CAMPBELL	WY	49	1968	2000	13.0	6987.0	6987.0	2214.0	.0	5920.0	17	0	0	0	.0	.0	77.0	10860.0	.00	21.0	107.0	.00	.67	.33	.0	.000
10713604507	0	PINEVIEW	50797505001 SUMMIT	UT	43	1975	2000	70.0	9339.0	9339.0	4225.0	.0	1600.0	10	0	0	0	.0	.0	1678.0	16164.0	.00	10.5	50.0	.00	.66	.34	.0	.000
10712809192	0	LEDGER	50097403001 PONDERA	MT	25	1974	2000	10.0	670.0	670.0	551.0	.0	39040.0	48	0	0	0	.0	.0	2139.0	7800.0	.00	22.0	189.3	.00	.65	.35	.0	.000
10713604505	0	ANSCHUTZ RANCH	50797803002 SUMMIT	UT	43	1978	2000	200.0	7036.0	7036.0	3111.0	.0	3800.0	4	0	0	0	.0	.0	3599.0	27600.0	.00	.0	37.0	.00	.78	.22	.0	.000
10713307185	0	BELL CREEK	51596701001 POWDER RIVER	MT	25	1967	2000	25.0	4400.0	4400.0	1190.0	.0	17600.0	61	0	0	0	.0	.0	20.0	36690.0	.00	24.0	1040.0	.00	.80	.20	.0	.000
10713402189	0	ELK BASIN	52091501102 CARBON	MT	25	1915	2000	190.0	4300.0	4300.0	2264.0	.0	1880.0	48	0	0	0	.0	.0	798.0	5748.0	.00	12.0	63.5	.00	.98	.02	.0	.000
10712101506	0	LISBON	58596001001 SAN JUAN	UT	43	1960	2000	225.0	8500.0	8500.0	2982.0	.0	5120.0	18	0	0	0	.0	.0	116383.0	147450.0	.00	5.5	22.0	.00	.71	.29	.0	.000
12710407279	0	MIST	71097901001 COLUMBIA	OR	9900	1979	2000	12.0	2448.0	2448.0	895.0	.0	2880.0	16	0	0	0	.0	.0	22223.0	72000.0	.00	25.0	200.0	.00	.70	.30	.0	.000
08713101240	0	ELKHORN RANCH	39596101001 BILLINGS	ND	33	1961	2000	8.0	9410.0	9410.0	4291.0	.0	6640.0	62	0	0	0	.0	.0	3035.0	16590.0	.00	16.0	3.8	.00	.64	.36	.0	.000
08713101242	0	FOOTHILLS, NORTHEAST	39595903001 BURKE	ND	33	1959	2000	15.0	6631.0	6631.0	2939.0	.0	5520.0	53	0	0	0	.0	.0	3296.0	19710.0	.00	12.0	3.8	.00	.75	.25	.0	.000
08713101233	0	ANTELOPE	39595301003 MC KENZIE	ND	33	1953	2000	21.0	10728.0	10728.0	5047.0	.0	6720.0	33	0	0	0	.0	.0	1007.0	7860.0	.00	10.1	4.3	.00	.65	.35	.0	.000
08713106243	0	FRYBURG	39595305001 BILLINGS	ND	33	1953	2000	10.0	8150.0	8150.0	3470.0	.0	11360.0	67	0	0	0	.0	.0	2785.0	14820.0	.00	11.8	20.0	.00	.79	.21	.0	.000
08713101249	0	RIVAL	39595715001 BURKE	ND	33	1957	2000	18.0	6067.0	6067.0	2760.0	.0	8640.0	36	0	0	0	.0	.0	2180.0	26250.0	.00	8.8	6.8	.00	.71	.29	.0	.000
08713106239	0	DICKINSON & WEST	39595705001 STARK	ND	33	1957	2000	13.0	7786.0	7786.0	3475.0	.0	10120.0	24	0	0	0	.0	.0	251.0	7230.0	.00	8.6	10.0	.00	.78	.22	.0	.000
08713101250	0	ROUGH RIDER	39595907001 MC KENZIE	ND	33	1959	2000	25.0	9300.0	9300.0	4400.0	.0	5440.0	65	0	0	0	.0	.0	3837.0	13920.0	.00	17.1	15.0	.00	.65	.35	.0	.000
08713101244	0	FRYBURG	39595305002 BILLINGS	ND	33	1953	2000	17.0	9403.0	9403.0	4170.0	.0	11360.0	67	0	0	0	.0	.0	896.0	6870.0	.00	16.0	3.8	.00	.68	.32	.0	.000
08713101235	0	BEAVER LODGE AREA	39595101002 WILLIAMS	ND	33	1951	2000	17.0	10490.0	10490.0	4800.0	.0	26320.0	98	0	0	0	.0	.0	8091.0	49350.0	.00	12.4	48.5	.00	.86	.14	.0	.000
08713101237	0	CHARLSON	39595201001 MC KENZIE	ND	33	1952	2000	77.0	8914.0	8914.0	4100.0	.0	10160.0	62	0	0	0	.0	.0	2966.0	39000.0	.00	4.2	3.8	.00	.62	.38	.0	.000
07716110503	0	WALNUT BEND	35093801002 COOKE	TX	4210	1938	2000	11.0	3450.0	3450.0	1885.0	.0	4600.0	154	0	0	0	.0	.0	87.0	5460.0	.00	21.0	138.0	.00	.66	.34	.0	.000
07714736342	0	FRANKS	22095529003 GALVESTON	TX	4210	1955	2000	28.0	8940.0	8940.0	3092.0	.0	1360.0	16	0	0	0	.0	.0	295.0	6510.0	.00	28.0	4000.0	.00	.88	.12	.0	.000
07714739392	0	BALDWIN	22093503001 NUECES	TX	4210	1935	2000	19.0	3814.0	3814.0	1825.0	.0	3000.0	9	0	0	0	.0	.0	2.0	7812.0	.00	32.0	460.0	.00	.74	.26	.0	.000
07714737344	0	GIST	22094728001 NEWTON	TX	4210	1947	2000	35.0	7079.0	7079.0	3349.0	.0	1040.0	21	0	0	0	.0	.0	21.0	7410.0	.00	32.2	2642.0	.00	.75	.25	.0	.000
07714725406	0	HAGIST RANCH	22094832003 DUVAL	TX	4210	1948	2000	9.0	5129.0	5129.0	2213.0	.0	10000.0	76	0	0	0	.0	.0	6.0	8796.0	.00	22.1	158.0	.00	.65	.35	.0	.000
07714833450	0	CAYUGA, NORTHWEST	26095303001 HENDERSON	TX	4210	1953	2000	45.0	7408.0	7408.0	3389.0	.0	2720.0	3	0	0	0	.0	.0	53.0	5804.0	.00	15.6	154.0	.00	.83	.17	.0	.000

.

.

.

(continues with other reservoirs)

Description of File: STOUND.STO

Data Element	Description	Format
1	11-character reservoir ID (Storage ID)	1x,a11
2	3-digit company code	1x,i3
3	field name	1x,a30
4	reservoir name	1x,a33
5	2-digit state code	1x,a2
6	4-digit state ID	1x,i4
7	discovery year (year)	1x,i4
8	year activated for storage (year)	1x,i4
9	pay thickness (feet)	1x,f6.1
10	maximum depth of the reservoir (feet)	1x,f7.1
11	minimum depth of the reservoir (feet)	1x,f7.1
12	original pressure (psig)	1x,f6.1
13	approximate acreage reservoir limit (acre)	1x,f7.1
14	approximate acreage total (acre)	1x,f7.1
15	number of output and/or input wells	1x,i5
16	number of pressure control and/or observation wells	1x,i5
17	number of compressors	1x,i4
18	horsepower of compressors (hp)	1x,i5
19	base gas total volume (MMCF)	1x,f8.1
20	working gas total volume (MMCF)	1x,f8.1
21	total future undeveloped/unused capacity (MMCF)	1x,f8.1
22	ultimate storage capacity (MMCF)	1x,f8.1
23	designed maximum deliverability (MMCF/D)	1x,f7.2
24	porosity (%)	1x,f4.1
25	permeability (md)	1x,f6.1
26	oil saturation (fraction)	1x,f4.2
27	gas saturation (fraction)	1x,f4.2
28	water saturation (fraction)	1x,f4.2
29	gas api gravity (°API)	1x,f4.1
30	gas specific gravity (dimensionless)	1x,f5.3

Table F-3

Input Data File: AFE.DAT (Location: \SRPM\DATA)
This file contains data for authorization for expenditure charges (not currently implemented).

```
C*** AFE Proportions
    Category                      % of Total
C****************************       C*****
Contractor Charges                  30.7
Road & Site Prep                    4.4
Transportation                      2.4
Fuel                                0.6
Mud & Additives                     5.8
Drillsite Logs & Monitoring         1.1
Other Physical Test                 0.6
Logs & Wireline Evaluations         3.7
Wellsite Data Services              0.1
Directional Drilling Services       1.6
Perforating                         1.4
Formation Treatment                 4.4
Cement & Services                   4.8
Casing & Tubing                     13.7
Special Tool Rentals                2.7
Drill Bits & Reamers                2.3
Wellhead Equipment                  1.8
Other Equipment & Supplies          3.0
Plugging                            1.3
Supervision & Overhead              5.6
Other Expenditures                  7.1
```

Table F-4

Input Data File: COST.DAT (Location: \ SRPM\DATA)
This file contains regional and resource-specific costs and investment data.

```
C*** Discount  Rate(%)
10.0
C*** Number of Cases
1
C*** Name of Case One
Current Technology
C*** Exploratory Well Costs Factor (multiplied with DWC to get EWC)
0.0
C*** Lease Bonus Cost Factor (multiplied with total revenues to get Lease Bonus)
0.005
C*** G&G Factor (Portion of EWC that is G&G costs)
0.05
C*** Development Dry Hole Costs as % of Total Development Well Costs (%)
70.0
C*** Percent Exploratory Well Cost Tangible (%)
25.0
C*** Percent Development Well Cost Tangible (%)
40.0
C*** Percent Facilities Cost Tangible (%)
100.00
C*** Environmental Capital Cost Multiplier (scaler of Facilities)
0.10
C*** G&A Expense Multiplier (scalar)
0.10
C*** G&A Capital Multiplier (scalar)
0.05
C*** Number of Regions (Excluding Default - 99)
20
C*** Development Well Cost (Function of Well Depth)
  1      60.1423     0.0         -4.85988e-7 9.73094e-10   2.4
  2      255.6081    0.0         -2.26601e-6 7.16859e-10   2.4
  3      90.5115     0.0          1.50565e-6 1.30923e-9    2.4
  4      19.2745     0.0          1.01848e-5  0.0          2.4
  5      218.1903    0.0          0.0        7.95466e-10   2.4
  6      71.6566     0.0         -5.21476e-7 1.01964e-9    2.4
  7      173.6535    0.0          2.94720e-6 1.81271e-10   2.4
  8      100.4499    0.0         -6.40013e-7 6.13326e-10   2.4
  9      108.4870    0.0          1.26400e-5  0.0          2.4
 10      137.7676    0.0          4.57525e-6 2.52925e-10   2.4
 11      52.8782     0.0         -1.54745e-6 4.10983e-9    2.4
 12      96.5663     0.0         -1.01096e-5  1.37204e-9   2.4
 13      915.5513    0.0          5.10503e-5 -2.38285e-9   2.4
 14      255.6081    0.0         -2.26601e-6 7.16859e-10   2.4
 15      915.5513    0.0         -2.04003e-5  2.38285e-9   2.4
 16      915.5513    0.0         -2.04003e-5  2.38285e-9   2.4
 17      491.4947    0.0         -2.16027e-5  2.51209e-9   2.4
 18      1000.       0.0         -5.21476e-7 1.01964e-9    2.4
 19      300         0.0         -5.21476e-7 1.01964e-9    2.4
 22      52.8782     0.0         -1.54745e-6 4.10983e-9    2.4
 99      100         0.0         -6.40000e-7 6.10000e-10   2.4
c regions
0
C**Environmental Costs
 99   0       0       0       0       0       0       0       0       0
C*** Facilities Well Cost (function of flow potential Mcf/Day)
C*** Number of Steps
1                                             Marginal
Max Mcf/Day      $/Well    $/(well*Mcf/Day)
C---------     C---------    C----------
12000.0          2500.0        25.00
C*** STMFAC(ITECH) Value, fraction
0.60
C*** Variable O&M Water ($/Barrel)
0.000
C*** Variable O&M Gas ($/Mcf including Compression) +Incremental of per 1000 feet depth
0.007  0.002
C**  Enter number of regions for fuel and shrinkage
14
```

```
C*** Enter region specific fuel and shrinkage (region # Fraction)
01   0.00
02   0.0222
03   0.0228
04   0.00
05   0.0138
06   0.01160
07   0.0159
08   0.0171
09   0.00
10  0.0101
11  0.02
12  0.02
13  0.00
14  0.00
C*** Enter default fuel and shrinkage (99  fraction)
99  0.010
C*** Annual Fixed O&M Well Cost (function of well depth)
C*** Number of Regions (Excluding Default - 99)
8
C-  C-    Region and Number of Steps
01  1                                            Marginal
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         5000.0        0.50
C-  C-    Region and Number of Steps
02  1                                            Marginal
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         6000.0        0.75
C-  C-    Region and Number of Steps
03  1                                            Marginal
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         5000.0        0.75
C-  C-    Region and Number of Steps
04  1                                            Marginal
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         5000.0        0.75
C-  C-    Region and Number of Steps
05  1                                            Marginal
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         5000.0        0.75
C-  C-    Region and Number of Steps
06  1                                            Marginal
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         5000.0        0.75
C-  C-    Region and Number of Steps
07  1                                            Marginal
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         5000.0        0.75
C-  C-    Region and Number of Steps
10  1                                            Marginal
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         5000.0        0.75
C-  C-    Region and Number of Steps
99  1                                            Marginal
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         5000.0        0.75
C*** Name of Technology two
Advanced Technology
C*** Exploratory Well Costs Factor (multiplied with DWC to get EWC)
1.2
C*** Lease Bonus Cost Factor (multiplied with total revenues to get Lease Bonus)
0.005
C*** G&G Factor (Portion of EWC that is G&G costs)
0.05
C*** Development Dry Hole Costs as % of Total Development Well Costs (%)
```

```
70.0
C*** Percent Exploratory Well Cost Tangible (%)
25.0
C*** Percent Development Well Cost Tangible (%)
40.0
C*** Percent Facilities Cost Tangible (%)
100.00
C*** Environmental Capital Cost Multiplier (scaler of Facilities)
0.10
C*** G&A Expense Multiplier (scalar)
0.10
C*** G&A Capital Multiplier (scalar)
0.05
C*** Number of Regions (Excluding Default - 99)
20
C*** Development Well Cost (Function of Well Depth)
  1      60.1423     0.0         -4.85988e-7 9.73094e-10  1.1
  2      255.6081    0.0         -2.26601e-6 7.16859e-10  1.1
  3      90.5115     0.0          1.50565e-6 1.30923e-9   1.1
  4      19.2745     0.0          1.01848e-5  0.0         1.1
  5      218.1903    0.0          0.0        7.95466e-10  1.1
  6      71.6566     0.0         -5.21476e-7 1.01964e-9   1.1
  7      173.6535    0.0          2.94720e-6 1.81271e-10  1.1
  8      100.4499    0.0         -6.40013e-7 6.13326e-10  1.1
  9      108.4870    0.0          1.26400e-5  0.0         1.1
 10      137.7676    0.0          4.57525e-6 2.52925e-10  1.1
 11      52.8782     0.0         -1.54745e-6 4.10983e-9   1.1
 12      96.5663     0.0         -1.01096e-5  1.37204e-9  1.1
 13      915.5513    0.0          5.10503e-5 -2.38285e-9  1.1
 14      255.6081    0.0         -2.26601e-6 7.16859e-10  1.1
 15      915.5513    0.0         -2.04003e-5  2.38285e-9  1.1
 16      915.5513    0.0         -2.04003e-5  2.38285e-9  1.1
 17      491.4947    0.0         -2.16027e-5  2.51209e-9  1.1
 18      1000.       0.0         -5.21476e-7 1.01964e-9   1.1
 19      300         0.0         -5.21476e-7 1.01964e-9   1.1
 22      52.8782     0.0         -1.54745e-6 4.10983e-9   1.1
 99      100         0.0         -6.40000e-7 6.10000e-10  1.1
C**Environmental Costs Number of Regions (or states)
0
CExt Tang  Intang  $/w/y  New Tang  Intang  $/w/y   $/foot  $/mcd  $/bblw
99   0       0       0       0       0       0       0       0       0
C*** Facilities Well Cost (function of flow potential Mcf/Day)
C*** Number of Steps
1                                              Marginal
Max Mcf/Day      $/Well    $/(well*Mcf/Day)
C---------     C---------    C----------
12000.0          200.0       20.00
C*** STMFAC(ITECH) Value, fraction
0.90
C*** Variable O&M Water ($/Barrel)
0.25
C*** Variable O&M Gas ($/Mcf including Compression) +Incremental of per 1000 feet depth
0.09   0.01
C**  Enter number of regions for fuel and shrinkage
13
C*** Enter region specific fuel and shrinkage (region #   fraction)
1   0.01
2   0.02
3   0.03
4   0.04
5   0.05
6   0.06
7   0.07
8   0.08
9   0.09
10  0.10
11  0.11
12  0.12
13  0.13
C*** Enter default fuel and shrinkage (99  fraction)
99  0.015
C*** Annual Fixed O&M Well Cost (function of well depth)
C*** Number of Regions (Excluding Default - 99)
4
```

```
C-  C-    Region and Number of Steps
01  1                                              Marginal
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0         1500.0        0.40
C-  C-    Region and Number of Steps
15  1                                              Marginal
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0        50000.0        1.25
C-  C-    Region and Number of Steps
16  1                                              Marginal
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0        40000.0        1.20
C-  C-    Region and Number of Steps
17  1                                              Marginal
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0        40000.0        1.20
C-  C-    Region and Number of Steps
99  1                                              Marginal
Max. Depth       $/Well      $/(Well-ft)
C---------     C---------    C----------
 15000.0          5000.0       1.20
```

Table F-5

Input Data File: DWLSPAC.DAT (Location: \ SRPM\DATA)
This file contains well spacing by storage/demand region. The data is to be used as a default value if it is not given in the SRPM database.

```
Code             Well Spacing For                   Region Name
                 Storage Fields (Used Only
                 When DB Doesn't Have Value)
                 (acre)
01                 20.0                             New England
02                 20.0                             Middle Atlantic
03                 20.0                             South Atlantic
04                 20.0                             Florida
05                 20.0                             East South Central
06                 160.0                            East North Central
07                 160.0                            West South Central
08                 20.0                             West North Central
09                 160.0                            Mountain 1
10                 160.0                            Mountain 2
11                 160.0                            California
12                 160.0                            Pacific Northwest
13                 160.0                            Canada-East
14                 160.0                            Canada-West
```

Table F-6

Input Data File: GEOLOGY.DAT (Location: \ SRPM\DATA)
This file specifies reservoir properties by pay-grade distribution (currently only one pay grade is considered in SRPM).

```
C*** Number of Reservoir Types  (Excluding Default)
2
C*** Pay Grade Parameters for Reservoir Type (Last is Default - 99)
Restype    P.G.   Acreage   Porosity    Netpay   H2O Saturation  Permeability
   7        1      0.00      1.0       1.0000      1.0            1.00
   7        2      1.00      1.0       1.0000      1.0            1.00
   7        3      0.00      1.0       1.0000      1.0            1.00
   9        1      0.00      1.0       1.0000      1.0            1.00
   9        2      1.00      1.0       1.0000      1.0            1.00
   9        3      0.00      1.0       1.0000      1.0            1.00
   99       1      0.00      1.0       1.0000      1.0            1.00
   99       2      1.00      1.0       1.0000      1.0            1.00
   99       3      0.00      1.0       1.0000      1.0            1.00
```

Table F-7

Input Data File: LEV.DAT (Location: \ SRPM\DATA)
This file contains levelized investment costs and fixed and variable O&M costs by operating company for existing storage reservoirs.

```
all costs in $/MCF of WG
      Levelized    Fixed Variable
Code  Inv Costs     O&M     O&M    Company
   1   0.616153 0.006200 0.024000  ANR Pipeline
   2   0.484606 0.016000 0.030000  ANR Storage
   5   0.452003 0.054400 0.056400  Arkla Energy Resources
  11   0.414477 0.052300 0.067800  CNG Transmission
  21   0.414477 0.052300 0.067800  Consolidated Gas Supply Corp.
  18   0.642328 0.025775 0.027200  Colorado Interstate Gas
  19   0.418490 0.018200 0.003200  Columbia Gas Transmission
  25   1.622263 0.013003 0.000000  El Paso
  27   0.528628 0.012800 0.015980  Equitrans
  45   1.233489 0.002400 0.022600  Michigan Gas Storage
  47   0.460125 0.010600 0.029800  Tennessee Gas Pipeline
  49   0.249978 0.006600 0.063400  Mississippi River Transmission
  54   0.487714 0.026376 0.000000  Questar
  56   0.767785 0.060600 0.028000  National Fuel Gas
  59   0.392028 0.045600 0.030000  Natural Gas Pipeline Co. of America
  69   0.767785 0.060600 0.028000  Penn-York Energy Corp.
  77   0.531265 0.092800 0.030400  Sonat
  78   0.568358 0.003000 0.025000  Southwest Gas
  79   0.568358 0.003000 0.025000  Southwest Gas
  82   0.213317 0.038800 0.024400  Texas Eastern
  83   0.703987 0.022800 0.033000  Texas Gas Transmission
  89   0.472911 0.033700 0.092800  Transco
  91   0.634921 0.020600 0.000000  Trunkline
 100   0.602646 0.043800 0.069400  Williams Natural Gas
 101   0.716344 0.032960 0.014160  Williston Basin
 107   0.452000 0.040000 0.038000  Blue Lake Gas Storage Company
 999   0.414477 0.052300 0.067800  CNG Transmission
```

Table F-8

File: PLAYINFO.DAT (Location: \ SRPM\DATA)
This file contains concentration of gas impurities by play.

```
USGS  H2S(%)  CO2(%)   N2(%)
2050  0.0000 10.0000  0.0000
2052  0.0000 10.0000  0.0000
2053  0.0000 25.0000  0.0000
2054  0.0000  4.5000  0.3000
2056  0.0000  6.5000  0.0000
2201  0.0000  1.2900  1.0520
2205  0.0000  0.8700  0.9600
2207  0.0000  0.8450  1.5500
2209  0.0054  0.4300  1.3640
2211  0.0000  0.8200  1.0500
2250  0.0000  1.7220  0.6710
2252  0.0000  0.9000  1.1000
3350  0.0000  0.5000  5.7000
4150  0.0000  0.6000  0.4000
4151  0.0000  0.6000  0.4000
4152  0.0000  0.6000  0.4000
4504  0.0005  0.4595  4.8110
4701  0.0095  0.2505  0.4960
4705  1.2883  6.7400  0.4780
4706  1.2879  6.7400  0.4545
4711  0.2284  5.2100  0.2500
4714  0.0000  0.0630  1.0370
4716  0.0000  0.0600  1.0650
4717  0.0000  0.1130  0.5970
4718  0.0000  2.0420  0.4660
4719  0.0000  2.8600  0.3525
4720  0.0000  2.8600  0.3650
4721  0.0000  2.8505  0.3315
4722  0.0000  2.8600  0.2880
4723  0.0000  2.7910  0.3390
4724  0.0000  0.8300  0.9850
4725  0.0000  0.8300  0.9630
4726  0.0001  1.1043  0.4593
4727  0.0001  1.2610  0.4315
4728  0.0000  0.2000  0.0290
4730  0.0125  0.2445  0.5290
4731  0.0125  0.2420  0.5210
4732  0.0056  0.1365  0.2335
4733  0.0042  0.3228  0.4518
4734  0.0042  0.3400  0.4835
4735  0.0052  0.3190  0.4860
4736  0.0042  0.2720  0.3940
4737  0.0042  0.2910  0.4550
4738  0.0289  0.2000  1.1800
4739  0.0049  0.0000  0.0000
4741  0.0542  0.1100  0.8700
4742  0.0542  0.1100  0.8700
4816  6.1625  5.3300  8.3700
4817  0.0320  5.3300  8.3700
4820  0.0716  2.6200  1.9000
4822  0.1185  2.2380  0.2460
4826  0.0000  0.9500  1.2310
4829  0.0260  1.1300  1.3400
4833  0.0049  1.6000  2.2820
4834  0.0000  0.5700  4.1000
4836  0.0000  1.2700  0.8100
```

.
.
.
(continues with other plays).

Table F-9

Input Data File: ROCKPROP.ADJ (Location: \ SRPM)
This file contains adjusted reservoir properties data for Technology Run for existing storage reservoirs (see STODIS.ADJ).

GSAMID	PERM (md)	POR (frac)	SG (frac)	NET PAY (ft)	SKIN	AREA (acres)	SPC (acres)	PEAK (MMCFD)
02706719218	5.122	.114	.610	13.124	12.000	1232.602	41.087	5.753
02706720448	2.417	.067	.638	86.249	12.000	746.146	37.307	170.639
02706721201	10.938	.102	.590	10.000	12.000	7943.218	69.071	48.163
02706721203	2.930	.101	.380	10.000	12.000	1651.554	330.311	2.943
02706721204	13.281	.107	.463	10.000	12.000	1115.830	557.915	5.089
02706721205	18.750	.107	.661	14.209	12.000	5770.929	115.419	277.446
02706721217	21.875	.097	.630	10.661	12.000	1529.931	218.562	36.044
02706721219	7.813	.115	.685	26.277	12.000	1373.367	42.918	145.297
02706721220	4.297	.114	.682	25.654	12.000	487.887	69.698	17.373
02706721221	3.516	.116	.688	27.113	12.000	853.937	37.128	49.669
02706721257	18.750	.102	.397	10.000	12.000	983.911	327.970	11.946
02706721258	11.719	.089	.604	20.582	12.000	2058.129	137.209	78.100
02706721259	18.750	.076	.558	12.842	12.000	711.189	237.063	12.493
02706721264	28.125	.180	.699	14.902	12.000	7954.264	94.694	840.222
02706721265	15.625	.104	.633	10.000	12.000	7468.495	162.359	496.194
02706721266	15.625	.105	.655	21.437	12.000	3478.407	77.298	363.925
02706721267	17.188	.111	.673	11.796	12.000	11309.040	176.704	890.597
02706721270	50.000	.145	.666	17.877	8.000	3962.880	141.531	492.391
02706721271	17.188	.135	.663	15.953	12.000	3772.198	94.305	263.744
02706721272	7.813	.110	.670	15.411	12.000	826.567	68.881	29.333
02706721287	9.375	.108	.663	13.029	12.000	4788.551	97.726	153.446
02706721296	3.516	.110	.645	12.192	12.000	1897.111	43.116	8.499
02706721304	20.313	.098	.633	23.002	12.000	1657.949	236.850	137.043
02706725268	8.594	.170	.746	146.177	12.000	6623.011	28.671	703.991
02706725275	8.594	.117	.576	10.000	12.000	1651.624	75.074	20.245
02706725300	6.375	.114	.684	30.372	12.000	408.663	58.380	5.686
02706725306	12.000	.110	.671	23.266	.000	376.483	75.297	9.975
02706725356	7.000	.086	.540	10.000	12.000	1680.380	37.342	12.602
02706732208	3.906	.094	.561	10.000	12.000	4183.734	66.408	9.222
02706732209	9.375	.094	.677	15.593	12.000	7504.073	47.195	99.084
02706732210	4.102	.097	.684	10.000	12.000	3398.443	75.521	11.053
02706732211	1.953	.082	.570	18.047	12.000	240.000	20.000	1.611
02706732212	4.492	.085	.603	10.000	12.000	1990.957	82.957	4.950
02706732213	5.078	.081	.565	10.000	12.000	2727.469	94.051	5.358
02706732214	5.078	.086	.650	11.531	12.000	4658.625	66.552	24.037
02706732215	3.906	.083	.639	10.377	12.000	2954.367	75.753	9.986
02706732216	4.688	.085	.644	10.870	12.000	2775.164	111.007	6.191
02706732282	1.758	.084	.461	10.000	12.000	838.174	69.848	1.980
02706737260	3.867	.172	.298	10.000	12.000	4806.198	42.912	43.192
02706737261	5.313	.106	.446	10.000	12.000	5705.496	150.145	10.589
02706737262	9.375	.090	.608	10.700	12.000	1155.702	88.900	4.649

.

.

.

(continues for other reservoirs)

Description of File: ROCKPROP.ADJ

Data Element	Description	Format
1	11-character reservoir ID (Storage ID)	2x,a11
2	adjusted permeability (md)	2x,f12.3
3	adjusted porosity (fraction)	2x,f12.3
4	adjusted gas saturation (fraction)	2x,f12.3
5	adjusted net pay thickness (ft)	2x,f12.3
6	adjusted skin factor	2x,f12.3
7	adjusted well drainage area (acres)	2x,f12.3
8	adjusted well spacing (acres)	2x,f12.3
9	maximum deliverability (MMCF/D)	2x,f12.3

Table F-10

Input Data File: SROM.TEM (Location: \ SRPM\DATA)
This file contains the template containing description of storage deliverability and injectivity and associated costs information for Demand and Integrating output files (*.SRO).

```
Dictionary
LC:   Levelized Investment Cost, $/MCF
FOM:  Fixed O&M Cost, $/MCF
VOM:  Variable O&M Cost, $/MCF
MERi: Maximum Extraction Rate for Season i, % of Working Gas Per Day
MIR:  Maximum Injection Rate for Season 4, % of Working Gas Per Day

storage id  storage   Total   Tot.   Yr.  Fuel   =========== OPTION 1 PARAMETERS ============    ======== OPTION 2 PARAMETERS =========     ===== OPTION 3 PARAMETERS =====   Storage Region
             first     W.G.   Norm.       Used    LC     FOM   VOM    MER1  MER2   MER3   MIR     LC     FOM    VOM   MER1   MER2   MIR      LC    FOM    VOM   MER1   MIR        Name
              Yr.     (MMCF)  B.G.      inj/ext
                                          (%)
```

Table F-11

Input Data File: TAX_NAT.DAT (Location: \ SRPM\DATA)
This file contains federal tax specifications and other national level tax structures and costs for U.S., must be copied to TAX_NAT.DAT if used for the analysis.

```
C*** Federal Income Tax Rate
34.0
C*** Independent Producer Depletion Rate (%)
100.0
C***  Are Intangible Drilling Costs to be Capitalized? (YES/NO)
YES
C***  Are Other Intangibles to be Capitalized? (YES/NO)
YES
C***  Include environmental Costs? (YES/NO)
YES
C***  Are Environmentals to be Capitalized? (YES/NO)
NO
C***  Implement Alternative Minimum Taxes? (YES/NO)
NO
C***  Allow AMT Taxes Paid to be Used as Credits in Future Years? (YES/NO)
YES
C***  Six Month Amortization Rate (%)
50.0
C***  Intangible Drilling Cost Preference Deduction (%)
100.0
C***  ACE Rate (%)
70.0
C***  Maximum Alternative Minimum Tax Reduction for Independents
0.0
C***  Alternative Minimum Tax RATE (%)
20.0
C***  Expense Environmental Costs? (YES/NO)
NO
C***  Allow Net Income Limitations? (YES/NO)
NO
C***  Net Income Limitation Limit (%)
40.0
C***  Percent Depletion Rate (%)
0.0
C***  Percent of Intan. Inv. to Capitalize (%)
30.0
C***  EOR Tax Credit Rate (%)
15.0
C***  Allow G&G Depletable Tax Credit? (YES/NO)
NO
C***  G&G Depletable Tax Credit Rate (%)
10.0
C***  Allow Tax Credit for Expensed G&G? (YES/NO)
NO
C***  G&G Intangible Tax Credit Rate (%)
15.0
C***  Allow Lease Acq. Depletable Tax Credit? (YES/NO)
NO
C***  Lease Acq. Depletable Tax Credit Rate (%)
10.0
C***  Allow Tax Credit for Expensed Lease Acq. Costs? (YES/NO)
NO
C***  Tax Credit Rate for Expensed Lease Acq. Costs (%)
15.0
C***  Allow Tangible Development Tax Credit? (YES/NO)
NO
C***  Tangible Development Tax Credit Rate (%)
15.0
C***  Allow Intangible Drilling Cost Tax Credit? (YES/NO)
NO
C***  Intangible Drilling Cost Tax Credit Rate (%)
15.0
C***  Allow Other Intangible Tax Credit? (YES/NO)
NO
C***  Other Intangible Tax Credit Rate (%)
15.0
C***  Allow Environmental Tangible Tax Credit? (YES/NO)
NO
C***  Environmental Tangible Tax Credit Rate (%)
20.0
C***  Allow Environmental Intangible Tax Credit? (YES/NO)
NO
C***  Environmental Intangible Tax Credit Rate (%)
20.0
C***  Allow Environmental Operating Cost Tax Credit? (YES/NO)
NO
C***  Environmental Operating Cost Tax Credit Rate (%)
20.0
C***  Allow Tax Credit On Tangible Investments? (YES/NO)
NO
C***  Number of Years for Tax Credit on Tangible Investments
20
C***  Allow Tax Credit On Intangible Investments? (YES/NO)
NO
C***  Number of Years for Tax Credit on Intangible Investments
15
```

```
C*** Regional Specific Taxes
C*** Royalty Rate (%)
12.5
C***  Percent of G&G Depleted (%)
16.17
C***  Allow  Forgiveness of State Taxes? (YES/NO)
NO
C***  Number of Years for Forgiveness of State Taxes
10
C***  Percent Lease Acquisition Cost Capitalized
100.0
```

Appendix F - Storage Reservoir Performance Module Files

(CONTINUED)

Table F-12

Input Data File: TAXES.DAT (Location: \ SRPM\DATA)
This file contains severance, income, and ad valorem taxes by State/District.

```
C*** State Tax Rates - Oil Severance Rates - Gas Severance Rates
C*** Gas values set to zero for storage module (4/22/96)
C*** Number of Regions (Excluding Default - 99)
59
Code     State   Oil     Oil    Gas      Gas
C*        (%)    (%)   ($/Bbl)   (%)    ($/MCF)
0100     0.00   10.00   0.000    0.00   0.00000  :Code| State Tax| Oil Sev. Tax| Gas Sev. Tax|
0105     5.00   10.00   0.000    0.00   0.00000  :0100:ALABAMA FED.OFFSHORE,0105:STATE OFFSHORE
0110     5.00   10.00   0.000    0.00   0.00000  :ALABAMA ONSHORE              :             :
5000     0.00   15.00   0.004    0.00   0.00000  :ALASKA SOUTH FED. OFFSHORE   :             :
5005     9.40   15.00   0.004    0.00   0.00000  :ALASKA SOUTH STATE OFFSHORE  :             :
5010     9.40   15.00   0.004    0.00   0.00000  :ALASKA SOUTH ONSHORE         :             :
5050     9.40   15.00   0.004    0.00   0.00000  :ALASKA NORTH ONSHORE         :             :
2        9.00    0.00   0.000    0.00   0.00000  :ARIZONA                      :             :
0310     6.50    5.00   0.000    0.00   0.00000  :ARKANSAS SOUTH               :             :
0350     6.50    5.00   0.000    0.00   0.00000  :ARKANSAS NORTH               :             :
0400     0.00    0.00   0.025    0.00   0.00000  :CALIFORNIA FED. OFFSHORE     :             :
0405     9.30    0.00   0.025    0.00   0.00000  :CALIFORNIA STATE OFFSHORE    :             :
0410     9.30    0.00   0.025    0.00   0.00000  :CALIFORNIA CENTRAL VALLEY    :             :
0450     9.30    0.00   0.025    0.00   0.00000  :CALIFORNIA COASTAL           :             :
0490     9.30    0.00   0.025    0.00   0.00000  :CALIFORNIA LOS ANGELES BASIN :             :
5        5.00    5.00   0.000    0.00   0.00000  :COLORADO                     :             :
0900     0.00    8.00   0.000    0.00   0.00000  :FLORIDA FED. OFFSHORE        :             :
0910     5.50    8.00   0.000    0.00   0.00000  :FLORIDA ONSHORE              :             :
12       4.80    0.00   0.000    0.00   0.00000  :ILLINOIS                     :             :
13       4.50    1.00   0.000    0.00   0.00000  :INDIANA                      :             :
15       4.00    8.00   0.000    0.00   0.00000  :KANSAS                       :             :
1700     0.00   12.50   0.000    0.00   0.00000  :LOUISIANA FED. OFFSHORE      :             :
1705     8.00   12.50   0.000    0.00   0.00000  :LOUISIANA STATE OFFSHORE     :             :
1710     8.00   12.50   0.000    0.00   0.00000  :LOUISIANA SOUTH              :             :
1750     8.00   12.50   0.000    0.00   0.00000  :LOUISIANA NORTH              :             :
21       2.30    6.60   0.000    0.00   0.00000  :MICHIGAN                     :             :
2300     0.00    6.00   0.000    0.00   0.00000  :MISSISSIPPI FED. OFFSHORE    :             :
2310     5.00    6.00   0.000    0.00   0.00000  :MISSISSIPPI ONSHORE          :             :
25       6.75    5.00   0.000    0.00   0.00000  :MONTANA                      :             :
26       7.81    3.00   0.000    0.00   0.00000  :NEBRASKA                     :             :
3010     7.60    7.09   0.000    0.00   0.00000  :NEW MEXICO SOUTHEAST         :             :
3050     7.60    7.09   0.000    0.00   0.00000  :NEW MEXICO NORTHWEST         :             :
33      10.50    5.00   0.000    0.00   0.00000  :NORTH DAKOTA                 :             :
3510     6.00    7.00   0.000    0.00   0.00000  :OKLAHOMA SOUTHWEST           :             :
3520     6.00    7.00   0.000    0.00   0.00000  :OKLAHOMA SOUTHEAST           :             :
3530     6.00    7.00   0.000    0.00   0.00000  :OKLAHOMA NORTHEAST           :             :
3540     6.00    7.00   0.000    0.00   0.00000  :OKLAHOMA NORTH CENTRAL       :             :
3550     6.00    7.00   0.000    0.00   0.00000  :OKLAHOMA NORTHWEST           :             :
40       6.00    4.50   0.000    0.00   0.00000  :SOUTH DAKOTA                 :             :
4200     0.00    4.60   0.000    0.00   0.00000  :TEXAS FED. OFFSHORE          :             :
4205     0.00    4.60   0.000    0.00   0.00000  :TEXAS STATE OFFSHORE         :             :
4210     0.00    4.60   0.000    0.00   0.00000  :TEXAS RRC DISTRICT 1         :             :
4220     0.00    4.60   0.000    0.00   0.00000  :TEXAS RRC DISTRICT 2         :             :
4230     0.00    4.60   0.000    0.00   0.00000  :TEXAS RRC DISTRICT 3         :             :
4240     0.00    4.60   0.000    0.00   0.00000  :TEXAS RRC DISTRICT 4         :             :
4250     0.00    4.60   0.000    0.00   0.00000  :TEXAS RRC DISTRICT 5         :             :
4260     0.00    4.60   0.000    0.00   0.00000  :TEXAS RRC DISTRICT 6         :             :
4270     0.00    4.60   0.000    0.00   0.00000  :TEXAS RRC DISTRICT 7B        :             :
4275     0.00    4.60   0.000    0.00   0.00000  :TEXAS RRC DISTRCIT 7C        :             :
4280     0.00    4.60   0.000    0.00   0.00000  :TEXAS RRC DISTRICT 8         :             :
4285     0.00    4.60   0.000    0.00   0.00000  :TEXAS RRC DISTRICT 8A        :             :
4290     0.00    4.60   0.000    0.00   0.00000  :TEXAS RRC DISTRICT 9         :             :
4295     0.00    4.60   0.000    0.00   0.00000  :TEXAS RRC DISTRICT 10        :             :
43       5.00    3.50   0.000    0.00   0.00000  :UTAH                         :             :
47       9.00    5.00   0.000    0.00   0.00000  :WEST VIRGINIA                :             :
49       0.00    6.00   0.000    0.00   0.00000  :WYOMING                      :             :
5300    15.50    5.00   0.000    0.00   0.00000  :CANADA  (Alberta)            :
5301    15.50    5.00   0.000    0.00   0.00000  :CANADA  (British Columbia)   :
5302    15.50    5.00   0.000    0.00   0.00000  :CANADA  (Saskatchewan)        :
9900     10.0    5.00   0.000    0.00   0.00000  :DEFAULT RATES                :             :
```

Table F-13

Data Input File: TECH.DAT (Location: \ SRPM\DATA)
This file contains technology parameters (only one (current) technology).

```
C*** Number of Technologies
1
C*** Name of Technology One
Current Technology
C*** Dry Hole Probability (%)
0.0
C*** Year to drill Infill wells for Water Drive Reservoirs
5.0
c**** number of regions for proration
14
c  Region Number and Proration (Fraction)
01  1.00
02  1.00
03  1.00
04  1.00
05  1.00
06  1.00
07  1.00
08  1.00
09  1.00
10  1.00
11  1.00
12  1.00
13  1.00
14  1.00
c   Default proration factor
99  1.00
c****number of states for state specific proration
0
c proration factors by state
c**** Number of different regions for Pay Continuity Enhancement
1
c  Pay Enhancement
1  1.0
c  Default for Pay Enhancement
99  1.0
c**** Number of different regions for System Pressure
6
c  Minimum system pressures by region
1    100.
2    100.
3    100.
4    100.
5    100.
6    100.
c  Default for Minimum System Pressure
99   100.
c  Number of Reservoir Types to Describe Well Performance Factors
9
c  Skin Factors for Reservoir Types 1 through Number above
6                    6                      6       6         6   6    6    6 6
c  Well Radius for Reservoir Types 1 through Number Above
0.650  0.650  0.650  0.650  0.650  0.650  0.65 0.65   0.65
c  Fracture Half Lengths for Reservoir Types 1 through Number above
0                    0                      0       0         0   0    0    0 0
c  Fracture Conductivity for Reservoir Types 1 through Number above
0                    0                      0       0         0   0    0    0 0
c number of regions for horizontal wells
0
c enter horizontal well info
c***** Number of different regions for tubing diameter
1
c ****** Enter tubing size by region (inches)
1   7.5
c ****** Enter tubing size default (inches)
99  7.5
c end of technology
```

Table F-14

Input Data File: TEMPLATE.DAT (Location: \ SRPM\DATA)

This file is a template file containing description of type curve input parameters, must be specified in REGIONS.DAT to create .TCI files.

```
ßßßßßßßßßßßßßßßßßß GSAM INPUT DATA FILE ßßßßßßßßßßßßßßßßßßßßßßßßßßßßßß
 CASE DESCRIPTION (2 Lines):
 MODULE6.INP - 10 md, Wet San Juan Basin Coal
 Input File for Module 6 Test Case
5ßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßß
                         IMPURITIES CONCENTRATIONS              SPEED UP
    GAS      TEMP.     ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ   TUBING ID    CASES
  GRAVITY   DEG. F       H2S       CO2       CN2      INCHES    1=Y,0=N
  ÄÄÄÄÄÄÄ   ÄÄÄÄÄÄ     ÄÄÄÄÄÄÄ   ÄÄÄÄÄÄÄ   ÄÄÄÄÄÄÄ   ÄÄÄÄÄÄÄÄÄ  ÄÄÄÄÄÄÄÄ
  0.6000     100.     0.00000   0.00000   0.00000     1.995       0
 ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ
                      BASIC RESERVOIR INFORMATION
                      ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ
  PAY    INITIAL  HORIZ  VERT    TOTAL   INIT WATER   NET PAY    WATER
 GRADE  PRESSURE  PERM.  PERM.  POROSITY SATURATION  THICKNESS  SALINITY
  NO.     PSIA     MD     MD    DECIMAL    DECIMAL     FEET       PPM
 ÄÄÄÄÄ  ÄÄÄÄÄÄÄÄ  ÄÄÄÄÄ  ÄÄÄÄÄ  ÄÄÄÄÄÄÄÄ ÄÄÄÄÄÄÄÄÄÄ  ÄÄÄÄÄÄÄÄÄ  ÄÄÄÄÄÄÄÄ
   1      2000.    10.   0.50   0.0100     1.00        40.0        0.
   2      2000.    10.   0.50   0.0100     1.00        40.0        0.
   3      2000.    10.   0.50   0.0100     1.00        40.0        0.
 ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ
       FRACTURED RESERVOIR INFORMATION
       ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ
  PAY    MATRIX    MATRIX   NAT'L FRAC
 GRADE    PERM.   POROSITY   SPACING
  NO.      MD     DECIMAL     FEET
 ÄÄÄÄÄ  ÄÄÄÄÄÄÄÄ  ÄÄÄÄÄÄÄÄ  ÄÄÄÄÄÄÄÄÄÄ
   1      5.00     0.1000      1.00
   2      5.00     0.1000      1.00
   3      5.00     0.1000      1.00
 ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ
                     FIELD DEVELOPMENT INFORMATION
                     ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ
                             WELL TYPE (MODULE NO.)  WELLBORE RADIUS, FT
  PAY                INITIAL ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ  ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ
 GRADE DEPTH  AREA   SPACING INITIAL FIRST  SECOND   INIT  FIRST  SECOND
  NO.   FEET  ACRES   ACRES   WELL   INFILL INFILL   WELL  INFILL INFILL
 ÄÄÄÄÄ ÄÄÄÄÄÄ ÄÄÄÄÄÄ ÄÄÄÄÄÄÄ ÄÄÄÄÄÄÄ ÄÄÄÄÄÄ ÄÄÄÄÄÄ  ÄÄÄÄÄÄ ÄÄÄÄÄÄ ÄÄÄÄÄÄ
   1    3500.  1280.   320.      6      6      6      0.33  0.33   0.33
   2    3500.  1280.   320.      6      6      6      0.33  0.33   0.33
   3    3500.  1280.   320.      6      6      6      0.33  0.33   0.33
 ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ
                   FRACTURED AND HORIZONTAL WELL DATA
                   ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ
       TYPE(VERT=0, HORIZ=1) FRAC Xf/HORIZ LENGTH     FRAC COND, MDÄFT
  PAY  ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ   ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ
 GRADE INIT  FIRST  SECOND    INIT  FIRST  SECOND    INIT  FIRST  SECOND
  NO.  WELL  INFILL INFILL    WELL  INFILL INFILL    WELL  INFILL INFILL
 ÄÄÄÄÄ ÄÄÄÄÄ ÄÄÄÄÄÄ ÄÄÄÄÄÄ    ÄÄÄÄÄ ÄÄÄÄÄÄ ÄÄÄÄÄÄ    ÄÄÄÄÄ ÄÄÄÄÄÄ ÄÄÄÄÄÄ
   1     0      0      0       500      0      0   100000      0      0
   2     0      0      0       500      0      0     1000      0      0
   3     1      0      0       500      0      0        0      0      0
 ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ
              WATER DRIVE AND UNCONVENTIONAL RESERVOIR DATA
              ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ
      AQUIFER       MAX
      Re/Rw  TRAP  WATER    0=DRY COAL LOCATION
 PAY  0: 2.5 GAS  >1 BPD/WL 1=WET COAL 0=APPAL.  GAS       LANG    DESOR
GRADE 1:  5  SAT  0-1%INFLX 2=DRY SH.  1=ALA.  CONTENT     PRES    TIME    DENS
 NO.  2: INF DEC. <0 BPM    3=WET SH.  2=WEST. (SCF/T)    (PSIA)  (DAYS) (G/CC)
ÄÄÄÄÄ ÄÄÄÄÄÄ ÄÄÄÄ ÄÄÄÄÄÄÄÄÄ ÄÄÄÄÄÄÄÄÄÄ ÄÄÄÄÄÄÄ ÄÄÄÄÄÄ      ÄÄÄÄ     ÄÄÄÄ   ÄÄÄÄ
  1      0   0.2   100.         1       2      500.       300.      10.   1.32
  2      1   0.2   100          1       2      500.       300.      10.   1.32
  3      2   0.2   100.         1       2      500.       300.      10.   1.32
 ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ
                                            WELL CONTROL INFORMATION
```

```
                                                  ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ
          MAXIMUM     INFILL                        SKIN FACTORS      SKIN FOR
         RATE MCFD   DATE FOR     PAY      ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ AUTO-REFRAC
MIN WHP  OR %AOF IF WTR DRIVE    GRADE   INIT   FIRST  SECOND   INITIAL
 PSIA    1 OR LESS  RESERVOIRS    NO.    WELL   INFILL INFILL     WELL
ÄÄÄÄÄÄÄ  ÄÄÄÄÄÄÄÄÄ  ÄÄÄÄÄÄÄÄÄÄ  ÄÄÄÄÄ  ÄÄÄÄÄ ÄÄÄÄÄÄ ÄÄÄÄÄÄ ÄÄÄÄÄÄÄÄÄÄÄ
  100.    12000.        2.         1      0.0    0.0    0.0    -4.0
                                   2      0.0    0.0    0.0    -4.0
                                   3      0.0    0.0    0.0    -4.0
ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ
TIME STEP CONTROL
ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ
Prod Time     Injection Time       Total       Maximum Time
Step Size      Step Size         Time Steps        Years
  Years          Years
ÄÄÄÄÄÄÄÄÄ     ÄÄÄÄÄÄÄÄÄÄÄ        ÄÄÄÄÄÄÄÄÄÄÄ   ÄÄÄÄÄÄÄÄÄÄÄÄ
   0.4           0.6                80           40.0
84ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ
```

Table F-15

Data Input File: REGIONS.DAT (Location: \SRPM\DATA)

This file specifies the file names of storage databases (*.STO) to be run through the SRPM and YES/NO switches as indicators for creating specific report files.

```
C*** Reports to Print
Type Curve Input File (*.tci)           YES
Detailed Pro-forma (*.pro)              YES
Type Curve Output (*.tco)               YES
Reduced Form Proforma (*.prr)           YES
Net Present Value Summary (*.npv)       YES
Print *.prd File                        YES
Region                          Run Type Curve ??
C********C**(t10)************    C**(t34)
stodis                           YES
stound                           YES
```

Table F-16

Input Data File: SRPMSPEC.DAT (Location: \ SRPM)
This file contains SRPM run specifications such as name of directory for output files, type of run for existing storage reservoirs, number of years and maximum working gas capacity for potential storage reservoirs.

```
Name of directory for output files
test-out
Run-type flag for existing storage reservoir: 0-create ROCKPROP.ADJ, 1=read ROCKPROP.ADJ
0
Number of years for potential storage mode run
20
Maximum working gas capacity (fraction of OGIP) for potential storage mode run
0.8
```

Table F-17

OUTPUT File: STODIS.ADJ (Location: \ SRPM\[OUTPUT DIRECTORY)
This file contains the adjusted reservoir properties data generated from Non-Technology Run/Base Run for existing storage reservoirs; must be copied to \SRPM\ROCKPROP.ADJ if Technology Run is to be performed.

```
      GSAMID     PERM (md)    POR (frac)     SG (frac)  NET PAY (ft)          SKIN  AREA (acres)   SPC (acres)  PEAK (MMCFD)
 02706719218         5.122          .114          .610        13.124        12.000      1232.602        41.087         5.753
 02706720448         2.417          .067          .638        86.249        12.000       746.146        37.307       170.639
 02706721201        10.938          .102          .590        10.000        12.000      7943.218        69.071        48.163
 02706721203         2.930          .101          .380        10.000        12.000      1651.554       330.311         2.943
 02706721204        13.281          .107          .463        10.000        12.000      1115.830       557.915         5.089
 02706721205        18.750          .107          .661        14.209        12.000      5770.929       115.419       277.446
 02706721217        21.875          .097          .630        10.661        12.000      1529.931       218.562        36.044
 02706721219         7.813          .115          .685        26.277        12.000      1373.367        42.918       145.297
 02706721220         4.297          .114          .682        25.654        12.000       487.887        69.698        17.373
 02706721221         3.516          .116          .688        27.113        12.000       853.937        37.128        49.669
 02706721257        18.750          .102          .397        10.000        12.000       983.911       327.970        11.946
 02706721258        11.719          .089          .604        20.582        12.000      2058.129       137.209        78.100
 02706721259        18.750          .076          .558        12.842        12.000       711.189       237.063        12.493
 02706721264        28.125          .180          .699        14.902        12.000      7954.264        94.694       840.222
 02706721265        15.625          .104          .633        10.000        12.000      7468.495       162.359       496.194
 02706721266        15.625          .105          .655        21.437        12.000      3478.407        77.298       363.925
 02706721267        17.188          .111          .673        11.796        12.000     11309.040       176.704       890.597
 02706721270        50.000          .145          .666        17.877         8.000      3962.880       141.531       492.391
 02706721271        17.188          .135          .663        15.953        12.000      3772.198        94.305       263.744
 02706721272         7.813          .110          .670        15.411        12.000       826.567        68.881        29.333
 02706721287         9.375          .108          .663        13.029        12.000      4788.551        97.726       153.446
 02706721296         3.516          .110          .645        12.192        12.000      1897.111        43.116         8.499
 02706721304        20.313          .098          .633        23.002        12.000      1657.949       236.850       137.043
 02706725268         8.594          .170          .746       146.177        12.000      6623.011        28.671       703.991
 02706725275         8.594          .117          .576        10.000        12.000      1651.624        75.074        20.245
 02706725300         6.375          .114          .684        30.372        12.000       408.663        58.380         5.686
 02706725306        12.000          .110          .671        23.266          .000       376.483        75.297         9.975
 02706725356         7.000          .086          .540        10.000        12.000      1680.380        37.342        12.602
 02706732208         3.906          .094          .561        10.000        12.000      4183.734        66.408         9.222
 02706732209         9.375          .094          .677        15.593        12.000      7504.073        47.195        99.084
 02706732210         4.102          .097          .684        10.000        12.000      3398.443        75.521        11.053
 02706732211         1.953          .082          .570        18.047        12.000       240.000        20.000         1.611
 02706732212         4.492          .085          .603        10.000        12.000      1990.957        82.957         4.950
```

.

.

.

(continues with other reservoirs)

Description of File: STODIS.ADJ

Data Element	Description	Format
1	11-character reservoir ID (GSAM ID)	2x,a11
2	adjusted permeability (md)	2x,f12.3
3	adjusted porosity (fraction)	2x,f12.3
4	adjusted gas saturation (fraction)	2x,f12.3
5	adjusted net pay thickness (ft)	2x,f12.3
6	adjusted skin factor	2x,f12.3
7	adjusted well drainage area (acres)	2x,f12.3
8	adjusted well spacing (acres)	2x,f12.3
9	maximum deliverability (MMCF/D)	2x,f12.3

Table F-18

OUTPUT File: STODIS.SRO (Location: \ SRPM\[OUTPUT DIRECTORY)
This file contains the existing storage reservoir performance output to be used in Demand and Integrating Module.

```
Dictionary
LC:   Levelized Investment Cost, $/MCF
FOM:  Fixed O&M Cost, $/MCF
VOM:  Variable O&M Cost, $/MCF
MERi: Maximum Extraction Rate for Season i, % of Working Gas Per Day
MIR:  Maximum Injection Rate for Season 4, % of Working Gas Per Day

storage id  storage   Total   Tot.   Yr.  Fuel   =========== OPTION 1 PARAMETERS ============    ======== OPTION 2 PARAMETERS =========     ===== OPTION 3 PARAMETERS =====   Storage Region
             first     W.G.   Norm.        Used   LC     FOM    VOM    MER1  MER2   MER3   MIR     LC     FOM    VOM   MER1   MER2   MIR      LC    FOM    VOM   MER1   MIR        Name
              Yr.     (MMCF)  B.G.      inj/ext
                                          (%)
02706719218  1947     517.1   .000  20.  2.22   .93    .07    .03   1.05    .97    .77    .28    .84    .07    .03    .99    .77    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706720448  1991   11292.8   .000  20.  2.22   .50    .06    .08   1.43   1.19    .69    .28    .46    .06    .07   1.23    .69    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721201  1947    3843.8   .000  20.  2.22   .51    .02    .00   1.19   1.06    .74    .28    .46    .02    .00   1.08    .74    .28    .42    .02    .00    .83    .28 Middle Atlantic
02706721203  1947     305.4   .000  20.  2.22   .51    .02    .00    .91    .87    .81    .28    .46    .02    .00    .88    .81    .28    .42    .02    .00    .83    .28 Middle Atlantic
02706721204  1971     501.7   .000  20.  2.22   .51    .02    .00    .96    .91    .79    .28    .46    .02    .00    .92    .79    .28    .42    .02    .00    .83    .28 Middle Atlantic
02706721205  1957   18280.5   .000  20.  2.22   .50    .06    .08   1.44   1.20    .69    .28    .46    .06    .07   1.24    .69    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721217  1950    2691.7   .000  20.  2.22   .93    .07    .03   1.27   1.11    .72    .28    .84    .07    .03   1.13    .72    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706721219  1980    9318.0   .000  20.  2.22   .93    .07    .03   1.47   1.22    .68    .28    .84    .07    .03   1.26    .68    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706721220  1972    1560.9   .000  20.  2.22   .93    .07    .03   1.06    .97    .77    .28    .84    .07    .03    .99    .77    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706721221  1980    4042.2   .000  20.  2.22   .93    .07    .03   1.16   1.04    .74    .28    .84    .07    .03   1.06    .74    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706721257  1969     946.6   .000  20.  2.22   .50    .06    .08   1.20   1.07    .74    .28    .46    .06    .07   1.09    .74    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721258  1972    5814.5   .000  20.  2.22   .51    .02    .00   1.27   1.11    .72    .28    .46    .02    .00   1.14    .72    .28    .42    .02    .00    .83    .28 Middle Atlantic
02706721259  1972     920.0   .000  20.  2.22   .51    .02    .00   1.29   1.12    .72    .28    .46    .02    .00   1.15    .72    .28    .42    .02    .00    .83    .28 Middle Atlantic
02706721264  1963   52724.6   .000  20.  2.22   .50    .06    .08   1.50   1.23    .67    .28    .46    .06    .07   1.27    .67    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721265  1961   29631.2   .000  20.  2.22   .50    .06    .08   1.57   1.25    .66    .28    .46    .06    .07   1.30    .66    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721266  1953   20414.9   .000  20.  2.22   .50    .06    .08   1.67   1.31    .64    .28    .46    .06    .07   1.37    .64    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721267  1959   56232.4   .000  20.  2.22   .50    .06    .08   1.49   1.21    .68    .28    .46    .06    .07   1.26    .68    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721270  1951   23731.5   .000  20.  2.22   .50    .06    .08   1.92   1.42    .60    .28    .46    .06    .07   1.50    .60    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721271  1951   17659.4   .000  20.  2.22   .50    .06    .08   1.41   1.18    .69    .28    .46    .06    .07   1.22    .69    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721272  1948    2249.9   .000  20.  2.22   .50    .06    .08   1.24   1.09    .73    .28    .46    .06    .07   1.11    .73    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706721287  1953   11865.2   .000  20.  2.22   .93    .07    .03   1.23   1.08    .73    .28    .84    .07    .03   1.10    .73    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706721296  1959     800.5   .000  20.  2.22   .93    .07    .03   1.01    .94    .78    .28    .84    .07    .03    .95    .78    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706721304  1971    9443.9   .000  20.  2.22   .50    .06    .08   1.38   1.16    .70    .28    .46    .06    .07   1.20    .70    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706725268  1951   60338.7   .000  20.  2.22   .50    .06    .08   1.10   1.01    .76    .28    .46    .06    .07   1.02    .76    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706725275  1947    1642.1   .000  20.  2.22   .64    .02    .02   1.17   1.05    .74    .28    .58    .01    .02   1.07    .74    .28    .53    .01    .02    .83    .28 Middle Atlantic
02706725300  1947     521.1   .000  20.  2.22   .50    .06    .08   1.03    .96    .78    .28    .46    .06    .07    .97    .78    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706725306  1947     557.7   .000  20.  2.22   .50    .06    .08   1.53   1.22    .67    .28    .46    .06    .07   1.27    .67    .28    .41    .05    .07    .83    .28 Middle Atlantic
02706725356  1947     857.8   .000  20.  2.22   .51    .02    .00   1.39   1.17    .70    .28    .46    .02    .00   1.21    .70    .28    .42    .02    .00    .83    .28 Middle Atlantic
02706732208  1951     882.8   .000  20.  2.22   .93    .07    .03    .99    .93    .79    .28    .84    .07    .03    .94    .79    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706732209  1952    7367.1   .000  20.  2.22   .93    .07    .03   1.27   1.11    .72    .28    .84    .07    .03   1.14    .72    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706732210  1948    1073.1   .000  20.  2.22   .93    .07    .03    .98    .92    .79    .28    .84    .07    .03    .93    .79    .28    .77    .06    .03    .83    .28 Middle Atlantic
02706732211  1950     136.1   .000  20.  2.22   .93    .07    .03   1.12   1.02    .76    .28    .84    .07    .03   1.03    .76    .28
```

.

.

.

(continues with other reservoirs)

Table F-19

OUTPUT File: STOUND.SRO (Location: \ SRPM\[OUTPUT DIRECTORY)
This file contains the potential storage reservoir performance output to be used in Demand and Integrating Module.

```
Dictionary
LC:   Levelized Investment Cost, $/MCF
FOM:  Fixed O&M Cost, $/MCF
VOM:  Variable O&M Cost, $/MCF
MERi: Maximum Extraction Rate for Season i, % of Working Gas Per Day
MIR:  Maximum Injection Rate for Season 4, % of Working Gas Per Day

storage id  storage   Total   Tot.   Yr.  Fuel   =========== OPTION 1 PARAMETERS ============    ======== OPTION 2 PARAMETERS =========     ===== OPTION 3 PARAMETERS =====   Storage Region
             first     W.G.   Norm.        Used    LC     FOM    VOM    MER1  MER2   MER3   MIR     LC     FOM    VOM   MER1   MER2   MIR      LC    FOM    VOM   MER1   MIR        Name
              Yr.     (MMCF)  B.G.       inj/ext
                                           (%)
11710901030  2000    8327.9  .010  20.  2.00    .65    .00    .05   2.32  1.62    .51   .28    .59    .00    .04  1.74    .51   .28    .54   .00    .04    .83    .28 California
11710903036  2000   28254.6  .016  20.  2.00    .43    .00    .05   1.57  1.29    .65   .28    .39    .00    .05  1.34    .65   .28    .35   .00    .04    .83    .28 California
11710903038  2000   35193.7  .008  20.  2.00    .76    .00    .06   2.62  1.67    .48   .28    .69    .00    .05  1.82    .48   .28    .62   .00    .05    .83    .28 California
11710903031  2000    8266.1  .011  20.  2.00    .75    .00    .05   2.59  1.65    .49   .28    .68    .00    .04  1.80    .49   .28    .62   .00    .04    .83    .28 California
11710903044  2000   15676.5  .009  20.  2.00    .87    .00    .05   2.94  1.73    .45   .28    .80    .00    .05  1.92    .45   .28    .72   .00    .04    .83    .28 California
11711001052  2000   68633.2  .009  20.  2.00    .79    .00    .05   2.74  1.71    .46   .28    .72    .00    .05  1.88    .46   .28    .65   .00    .04    .83    .28 California
11711401061  2000  162438.7  .010  20.  2.00    .53    .00    .05   1.94  1.49    .57   .28    .48    .00    .04  1.56    .57   .28    .44   .00    .04    .83    .28 California
11711404056  2000   38081.5  .003  20.  2.00  18.40    .03    .05  18.82   .23    .00   .28  16.73    .02    .05  3.23    .00   .28  15.21   .02    .04    .83    .28 California
11711403060  2000   84004.2  .011  20.  2.00    .44    .00    .05   1.64  1.34    .63   .28    .40    .00    .05  1.39    .63   .28    .37   .00    .04    .83    .28 California
06716308133  2000   11737.2  .009  20.  1.16    .91    .03    .05   2.87  1.70    .46   .28    .83    .02    .05  1.89    .46   .28    .75   .02    .04    .83    .28 East North Central
06716307141  2000    9550.4  .009  20.  1.16    .87    .01    .06   2.92  1.71    .45   .28    .79    .00    .05  1.91    .45   .28    .72   .00    .05    .83    .28 East North Central
06716301123  2000   12982.0  .023  20.  1.16    .80    .12    .06   1.73  1.33    .63   .28    .73    .11    .05  1.40    .63   .28    .67   .10    .05    .83    .28 East North Central
06716308127  2000   15426.7  .014  20.  1.16   1.73    .34    .06   2.22  1.52    .55   .28   1.57    .31    .05  1.63    .55   .28   1.43   .28    .05    .83    .28 East North Central
06716307128  2000    9385.3  .009  20.  1.16    .89    .02    .06   2.94  1.69    .46   .28    .81    .01    .06  1.89    .46   .28    .74   .01    .05    .83    .28 East North Central
06716312134  2000    9850.9  .008  20.  1.16    .94    .00    .07   3.10  1.74    .44   .28    .85    .00    .06  1.96    .44   .28    .78   .00    .06    .83    .28 East North Central
06716307129  2000   14081.3  .009  20.  1.16    .84    .00    .06   2.82  1.69    .47   .28    .76    .00    .06  1.87    .47   .28    .70   .00    .05    .83    .28 East North Central
06716307139  2000   16916.8  .009  20.  1.16    .87    .00    .06   2.94  1.72    .45   .28    .79    .00    .05  1.91    .45   .28    .72   .00    .05    .83    .28 East North Central
06716307136  2000   10826.4  .008  20.  1.16    .96    .01    .06   3.15  1.73    .44   .28    .87    .01    .06  1.96    .44   .28    .80   .01    .05    .83    .28 East North Central
06716311115  2000  166476.0  .010  20.  1.16    .79    .01    .05   2.71  1.67    .48   .28    .72    .01    .05  1.84    .48   .28    .65   .01    .04    .83    .28 East North Central
05716502155  2000   10414.1  .008  20.  1.38   1.02    .02    .05   3.22  1.76    .42   .28    .93    .01    .05  2.00    .42   .28    .84   .01    .04    .83    .28 East South Central
05716502015  2000   12724.0  .009  20.  1.38    .88    .01    .06   2.92  1.71    .46   .28    .80    .01    .05  1.90    .46   .28    .72   .01    .05    .83    .28 East South Central
05716503012  2000    9893.0  .009  20.  1.38    .95    .02    .05   3.11  1.75    .43   .28    .86    .02    .05  1.97    .43   .28    .78   .02    .04    .83    .28 East South Central
05716502016  2000   10752.4  .009  20.  1.38    .90    .01    .06   3.01  1.73    .44   .28    .82    .01    .05  1.94    .44   .28    .74   .01    .05    .83    .28 East South Central
05716502172  2000   11274.8  .010  20.  1.38    .81    .01    .06   2.75  1.67    .48   .28    .74    .01    .05  1.84    .48   .28    .67   .01    .05    .83    .28 East South Central
05716503162  2000   39760.6  .009  20.  1.38    .98    .02    .05   3.19  1.76    .43   .28    .89    .02    .05  1.99    .43   .28    .81   .02    .04    .83    .28 East South Central
05714946158  2000   21108.4  .001  20.  1.38  26.17    .24    .06  18.80   .23    .00   .28  23.79    .22    .05  3.22    .00   .28  21.63   .20    .05    .83    .28 East South Central
05714943182  2000   16339.9  .002  20.  1.38   7.53    .13    .06  12.44  1.29    .05   .28   6.85    .12    .05  3.09    .05   .28   6.22   .11    .05    .83    .28 East South Central
05714940168  2000   16557.8  .002  20.  1.38  22.19    .31    .06  18.54   .28    .00   .28  20.17    .29    .05  3.23    .00   .28  18.34   .26    .05    .83    .28 East South Central
05714937157  2000   83638.6  .001  20.  1.38  28.44    .08    .07  19.10   .17    .00   .28  25.86    .07    .06  3.22    .00   .28  23.51   .06    .06    .83    .28 East South Central
04714910070  2000   39539.5  .009  20.   .00    .91    .00    .10   2.95  1.68    .46   .27    .82    .00    .09  1.88    .46   .27    .75   .00    .08    .83    .27 Florida
09712102018  2000    4489.9  .044  20.   .00   1.22    .11    .06   1.35  1.15    .70   .27   1.11    .10    .05  1.19    .70   .27   1.01   .09    .05    .83    .27 Mountain 1
09714406227  2000   10859.1  .010  20.   .00   1.05    .01    .08   3.07  1.66    .46   .27    .95    .01    .07  1.89    .46   .27    .86   .00    .07    .83    .27 Mountain 1
09719357230  2000   10238.9  .009  20.   .00    .93    .02    .07   2.96  1.70    .46   .27    .84    .02    .07  1.90    .46   .27    .76   .02    .06    .83    .27 Mountain 1
09712207208  2000   16315.1  .008  20.   .00   1.44    .06    .06   3.45  1.79    .40   .27   1.31    .05    .05  2.06    .40   .27   1.19   .05    .05    .83    .27 Mountain 1
09719357218  2000   11204.2  .010  20.   .00    .92    .01    .07   2.83  1.67    .47   .27    .83    .01    .07  1.85    .47   .27    .76   .01    .06    .83    .27 Mountain 1
09714410224  2000    5975.6  .003  20.   .00   5.56    .65    .05  10.22  1.60    .08   .27   5.05    .59    .05  2.99    .08   .27   4.59   .54    .04    .83    .27 Mountain 1
09712201212  2000   23624.0  .009  20.   .00    .91    .01    .06   2.99  1.70    .45   .27    .83    .01    .06  1.91    .45   .27    .75   .01    .05    .83    .27 Mountain 1
09714411219  2000   24260.9  .011  20.   .00   1.04    .06    .05   2.87  1.69    .46   .27    .95    .05    .05  1.88    .46   .27    .86   .05    .04    .83    .27 Mountain 1
09719357216  2000   19650.3  .010  20.   .00    .83    .00    .08   2.64  1.64    .49   .27    .76    .00    .07  1.80    .49   .27    .69   .00    .06    .83    .27 Mountain 1
09719357215  2000   42137.7  .008  20.   .00    .79    .00    .07   2.63  1.68    .48   .27    .71    .00    .07  1.83    .48   .27    .65   .00    .06    .83    .27 Mountain 1
```

.

.

.

(continues with other reservoirs)

APPENDIX G
ERROR MESSAGES

(EXPLORATION AND PRODUCTION MODULE AND DEMAND AND INTEGRATING MODULES)

APPENDIX G -- ERROR MESSAGES

101
Error in specification of time periods (1993, 1995…). Must be increasing with no blanks between year specifications. The years specified should correspond to years for which gas price is available in GASPRC.NEW file—check GEN_TML.SPC file for errors

102
Error in reading the *.PRD and *.DEC files. Check the files and MAKEBIN.FOR for binary "data bank" file creation (E&P)

103
Error in creating the binary "data bank" file. Check the *.PRD and *.DEC and *.GSM input files and MAKEBIN.FOR.

104
Too many supply regions specified. Check NODE.SPC file. The supply region counter should not be more than 24.

105
Same supply region specified twice. Check NODE.SPC file.

106
Too many demand regions specified. Check NODE.SPC file. The demand region counter should not be more than 16.

107
Demand region specified twice. Check NODE.SPC file.

108
No supply regions identified. At least one supply region should be specified in NODE.SPC file for the E&P Module.

109
No demand regions identified. At least one demand region should be specified in NODE.SPC file for the D&I Module.

110
A supply region has not been defined. Check NODE.SPC file.

111
A demand region has not been defined. Check NODE.SPC file.

112
Too many transport links have been specified. The total number of transportation links should not exceed 80. Check LINK_NDE.SPC file.

113
Origin or Destination node not found in node list. Check LINK_NDE.SPC file.

114
Year for the existing (first) pipeline capacity should be less than or equal to the year specified for pipeline capacity additions. Check LINK_NDE.SPC file.

115
No transport links specified

116
Storage option node specification not found

117
Too many storage options for specified node

118
Demand region specification (see above) does not match demand regions

119
Too many demand increments specified

120
Too many demand sectors specified

121
Demand load profile does not average out properly

122
Too many types of peaking supplies

123
No peaking supplies specified

124
No match on specified peaking supply or node name

125
Too many other supply projects

126
Specified node not found in supply project specifications

201
Too many supply increments or bad ordering of supply increment data

202
Bad ordering of supply pass indicator in supply specifications

203
Load factor (Peak/average) is less than 1.0

204
The specifications of the supply pass differs for the same supply increment

205
Node name not found in list of supply/demand regions

301
The number of matrix codes are insufficient for the number of options

401
Supply region name not found. Check GASPRC.NEW for supply region name

402
Supply region incorrectly specified in DRL_RCP.SPC file

403
Supply region not found for DRL_RCP.SPC file

404
All data in DRL_RCP.SPC file not read.

405
Number of plays specified in PLY_DFN.SPC file exceeds maximum allowed (currently maximum number of plays that could be specified is 995).

406
State specification is either '0' or greater than 50 in PLY_DFN.SPC.

407
Supply region not defined in GASPRC.NEW file.

410
Total number of exploration technology specifications exceeds maximum allowed. Check ETEC_PEN.SPC file.

410
Resource type out of range in ETEC_PEN.SPC file. Resource type specification is either zero or greater than 7.

411
Year of exploration technology out of range i.e. year specified for exploration technology is either less than 1993 or greater than 2033.

412
Error in initializing exploration technology rate of penetration

413
No exploration technologies specified. Check ETEC_PEN.SPC file.

414
Range of years for exploration technology exceed maximum. Check ETEC_PEN.SPC file.

415
Play name not found.

416
Mis-match between the name of exploration technology specified in ETEC_PEN.SPC file and EXP_DFN.SPC file.

417
Exploration specification previously set for reservoir class.

418
Exploration specification previously set for play. Check EXP_PLY.SPC file.

419
Number of Development technology specifications exceed maximum allowed. Check DTEC_PEN.SPC file

420
Year of development technology penetration out of range i.e. year specified for development technology is either less than 1993 or greater than 2033.

421
Error in initializing development technology rate of penetration.

422
Range of years for development technology exceed maximum. Check DTEC_PEN.SPC file.

424
Tax Codes specifications exceed maximum.

425
Year specified for royalty incentive out of bounds. Only one year for royalty incentive can be specified. Check TAX_CDE.SPC file.

426
Royalty incentive specification year is invalid. Check TAX_CDE.SPC file.

427
Royalty incentive year is less than 1993 or greater than 2033. Check TAX_CDE.SPC file.

428
Tax code incorrectly specified

428
Drilling tax specification incorrectly specified

429
Drilling tax specification not specified for tax code change

430
Exploration cost factors inconsistent

431
Input does not match development technology in DVL_TPR.SPC file.

432
Resource type indicator (1,2,3…7) is higher than maximum resource counter. Check DVL_TPR.SPC file.

433
Change in rig fleet not correctly set

451
Reservoir count exceeds maximum allowed total number of reservoirs (higher than 17,200). Reduce number of reservoirs.

451
Resource type is out of bounds. Check the ‘data bank’ file.

452
Supply region for *.PRD and *.DEC files not consistent. Check the ‘data bank’ file.

452
Field size class for *.PRD and *.DEC files not consistent. Check the ‘data bank’ file.

453
Play name in the ‘data bank’ does not match with any play name specified in PLY_DFN.SPC file.

454
Field size class is less than 5or greater than 17. Check the ‘data bank’.

455
Pay grade counter is either ‘0’ or greater than ‘3’ in ‘data bank’.

456
Technology name is incorrect in the ‘data bank’. (C for current tech., A for advanced)

457
Development option is incorrect in the ‘data bank’ (P for primary, R for re-frac, I for infill case).

458
Resource type is out of bounds in ‘data bank’.

459
Undiscovered accumulations (field size class in a play) in the ‘data bank’ greater than the maximum specified (6200).

461
Depth of reservoir incorrectly set in the E&P Module.

462
Secondary window not correctly set in the E&P Module.

463
Well calculation is not correct in RP Module. Check the ‘data bank’ for # of wells for primary drilling vs. infill drilling.

464
Initialization of NPV of production not done correctly in E&P Module.

465
Undiscovered accumulations (field size class in a play) in the ‘data bank’ greater than the maximum specified (6200).

466
Only one technology specified in ‘data bank’

467
Play name in the *.DEC and *.PRD files do not match.

468
Play mis-match in the ‘data bank’ and PLY_DFN.SPC files.

469
Total number of reservoirs in ‘data bank’ exceed maximum allowed.

470
Pay grade counter is either ‘0’ or greater than ‘3’ in the ‘data bank’.

471
Resource type specification out of bounds in the ‘data bank’.

472
Technology name is incorrect in the ‘data bank’. (C for current tech., A for advanced)

473
Development option is incorrect in the ‘data bank’ (P for primary, R for re-frac, I for infill case).

474
Resource type specified in the ‘data bank’ is not consistent with PLY_DFN.SPC file.

475
Depth of reservoir incorrectly set in the E&P Module.

476
Year for secondary development not consistent

477
Secondary MASP is out of bounds

478
Initialization of NPV of production not done correctly in E&P Module.

479
No discovered reservoirs in the ‘data bank’.

480
Number of wells in pay grade different for the two technologies modeled. Check the ‘data bank’

481
Number of discovered reservoirs exceeds maximum.

482
Number of undiscovered accumulations exceed maximum.

483
Number of undiscovered accumulations is zero in the ‘data bank’.

489
Input/output code not correctly set for binary data bank creation. Check IOCDE.SPC file.

490
Number of reservoirs in the *.PRD and *.DEC file exceeds maximum specified in the E&P Module.

491
Components of GSAM ID not consistent in *.PRD and *.DEC files.

493
Pay grade counter is either ‘0’ or greater than ‘3’ in ‘data bank’.

494
Resource type counter is not set correctly.

495
Technology name is incorrect in the ‘data bank’. (C for current tech., A for advanced)

496
Development option is incorrect in the ‘data bank’ (P for primary, R for re-frac, I for infill case).

497
Improper initialization of production stream.

501
Duplicate population/economic data exists in input databases in D&I Module.

501
Pointer for reservoirs improperly assigned in E&P Module.

502
Population/economic specifications are incomplete

503
Base and scenario population data are inconsistent for 1st year.

504
Base and scenario economic growth data are inconsistent for 1st year.

505
Residential price elasticity for specified demand region has wrong sign.

506
Specified demand region in residential demand specifications not found.

507
Residential demand specifications duplicated for specified demand region.

508
Residential demand specifications incomplete for specified demand region.

509
Commercial price elasticity for specified demand region has wrong sign.

510
Specified demand region in commercial demand specifications not found.

511
Commercial demand specifications duplicated for specified demand region.

512
Commercial demand specifications incomplete for specified demand region.

513
Specified demand region in industrial demand specifications not found.

514
Too many sub-sectors in industrial sector.

515
Industrial demand specifications duplicated for specified demand region
and sub-sector.

516
Industrial sharing factors do not add to 100% for specified region,
sub-sector and year.

517
Industrial demand specifications incomplete for specified demand region
and sub-sector.

518
Share of EU demand by load period does not add to 1.0 for specified
region, type, and fuel.

519
Specified demand region not found in EU specifications.

520
Duplicated total electricity sales data for specified region.

521
Incorrect fuel specified or duplicate existing capacity data in EU
data for specified region.

522
Incorrect data type flag in EU data (see above).

523
Incomplete specification of total sales data. No data for specified
region

524
Incomplete specification of existing EU capacity data for specified
region and fuel type (1- coal, 2- gas only, 3-oil only,
4- distillate/gas, 5- Low sulfur resid/gas, 6- High sulfur resid/gas)

525
Specified fuel type not found in EU capacity cost and efficiency data.

526
Duplicate fuel cost/efficiency specifications for specified fuel type.

527
EU generation efficiency/cost data not found for specified fuel type
(1- coal, 2- gas only, 3-oil only, 4- distillate/gas,
5- Low sulfur resid/gas, 6- High sulfur resid/gas)

528
Demand factor incorrectly set

601	End of file reached without finding reservoir
601	Number of years in run not specified or is higher than 40. Check *.PRD file for number of years specification.
602	Number of projects exceeds maximum projects allowed. Check EX_SZE.CMN.
603	Total reservoir count exceeds maximum
604	Error in development status of reservoir
605	Reservoirs not in correct order
605	Known reservoir count exceeds maximum
606	Response from only one technology store in the 'data bank'.
641	Undiscovered reservoir count not specified
642	Undiscovered reservoir count is less than zero.
701	Maximum tax codes exceeded
702	Technology case incorrectly set
711	No reservoirs to process. Check for availability of 'data bank'.
714	Reservoir count exceeds maximum
715	Number of development options exceeds maximum
721	Exploration technology specifications size class out of bounds
801	Drilling cost factor for reservoir not specified
802	Play designation not set
803	Drilling cost factor for reservoir not specified
804	Drilling cost factor out of bounds
805	Play description not found
806	Play description incorrect
807	Play description incorrect
811	Drilling cost factor not set
813	Drilling cost factor not set
861	No resource found for field class
862	Exploration factor not set for field/reservoir size class
871	Variable drilling cost factor set to zero
872	Technology drilling cost scaling factor not set, or drilling cost decline exceeds 100%

```
873
     Play designation not set for discovered play

881
     Variable drilling factor not set

882
     Technology drilling cost scaling factor not set, or drilling cost decline exceeds 100%

888
     Maximum resource in play set to zero

889
     Exploration cost factor not specified

912
     Technology not found for option  (open file: drl_tpr.spc)  FATAL ERROR

The NON-FATAL messages below indicate program is executing the following
section of code

913
     EXDVI1.FOR line 1089

914
     EXDVI2.FOR line 523

915
     EXDVI2.FOR line 845

921
     EXDVST.FOR line 471

922
     EXDVST.FOR line 678

923
     EXDVST.FOR line 839

924
     EXDVST.FOR line 1068

925
     EXDVST.FOR line 1094

926
     EXDVST.FOR line 1102

927
     EXDVST.FOR line 1534

928
     EXDVST.FOR line 1157

929
     EXDVST.FOR line 1667

930
     EXDVST.FOR line 1727

931
     EXDVST.FOR line 1905

932
     EXDVST.FOR line 1993

951
     INTMGN.FOR line 33

952
     INTMGN.FOR line 38

953
     INTMGN.FOR line 236

954
     INTMGN.FOR line 731

955
     INTMGN.FOR line 738

956
     INTMGN.FOR line 770
```

PROGRAMMER'S GUIDE FOR THE RESERVOIR PERFORMANCE (RP) MODULE OF THE GAS SYSTEMS ANALYSIS MODEL (GSAM)

FINAL REPORT

Volume IIIa – RP Programmer's Guide

For:

U.S. Department of Energy
National Energy Technology Laboratory
Morgantown, West Virginia
Under Contract Number: DE-AC21-92MC28138

By:

ICF Consulting, Inc.
Fairfax, Virginia

February 2001

Table of Contents

RESERVOIR PERFORMANCE PROGRAMMER'S GUIDE

This programmer's guide provides a detailed description of computer code for the Reservoir Performance (RP) Module of the Gas System Analysis Model (GSAM). The guide is divided into various sections. Section DATA DICTIONARY gives description of global variables used in the RP module and shows the location of the variables in header files ".H". Logical flow of subroutines of the RP module is given in section FLOW CHARTS.
The remaining sections of the programmer's guide describe the main program and subroutines of the RP module with detailed discussion and explanation of each step in the code.

Program RESVPERF.EXE

File RESVPERF.EXE is the executable program of the RP module. This program is a compilation of one main program, 25 header ".H" files (file holder for global variables), and 85 sub-programs (subroutines). Names of the main program and subroutines and their locations in the ".FOR" files are given in the program sections. Names of the header files are as follows:

CASHFLOW.H	TYPE1.H
COST.H	TYPE2.H
COSTING.H	TYPE3.H
DIMEN.H	TYPE4.H
FIELD.H	TYPE5.H
GEOLOGY.H	TYPE6.H
GLOBAL.H	TYPE7.H
GSAMVAR.H	TYPE8.H
NPV.H	TYPE9.H
TAX_NAT.H	TYPE10.H
TAX_REG.H	UNITCOST.H
TECH.H	WELLDATA.H
TYPE_OUT.H	

General Structure of the Program Sections

The explanation of each routine in the program section is started with the name of the routine. If there are parameters passed to the routine, extension "()" is added to the name of the routine. Before the explanation for the code begins, there are seven subheadings:

1. **LOCATION:**
   Gives name of the FORTRAN file that stores the routine.
2. **MAIN THEME:**
   Briefly describes the main purpose of the routine.

3. **CALLS:**
   List of routines (name, location, and brief description) that are invoked by the routine.
4. **CALLED BY:**
   List of routines (name, location, and brief description) that call the routine.
5. **READS:**
   List input files (name and brief description) read by the routine.
6. **CREATES:**
   List output files (name and brief description) created by the routine.
7. **ROUTINE INTERACTIONS:**
   Shows the interactions between the calling routines, the routine itself, and the invoked routines in the form of a flow chart. List of parameters passed to the routine (if any) is also given.

These subheadings are followed by detailed explanations for the computer code. Most of the code is explained in steps, i.e., the explanation for a section of related code is delegated in a single step. Between steps, if a certain section of code needs further explanation, a "**Note**" is inserted with the relevant explanation. All variables in the explanation are written in *italic*, and extension "*()*" is appended to the array variables.

Variable Name	Location	Description
aatcf	Cashflow.h	Annual After Tax Cash Flow
AbsRns	Type4.h	Absolute Roughness of Pipe
acdv93	Gsamvar.h	Reservoir Developed Area EOY 1993
ace	Cashflow.h	Adjusted AMT income
aceadj	Cashflow.h	ACE ADJustment
acer	Tax_nat.h	ACE Rate
acpamt	Cashflow.h	Availible Credits for Past AMT
acprod	Gsamvar.h	Estimated Total Production Area
acprov	Gsamvar.h	Maximum Proved Area
acprvd	Gsamvar.h	Date of Maximum Proved Area Estimate
Adesrb	Type8.h	Non-Equilibrium Amount of gas absorbed for Coal-bed Methane
adjgross	Cashflow.h	ADJusted GROSS sales
afe	Unitcost.h	AFE Proportions
amint	Cashflow.h	Alternative MINimum Taxes
amt	Tax_nat.h	Alternative Minimum Tax
amti	Cashflow.h	Alternative Minimum Taxiable Income
amtrate	Tax_nat.h	Altermative Minimum Tax RATE
apd	Cashflow.h	Allowable Percent Depletion
aquprm	Gsamvar.h	Aquifer Permeability
aqurad	Gsamvar.h	Aquifer Radius
Area	Type4.h	Total Reservoir Area by Pay Grade
area_fac	Geology.h	factor for pay grade acreage
avdep	Field.h	Average Well Depth
bamtp	Cashflow.h	Balance of AMT Paid
Bdesrb	Type8.h	Constant used in calculation of non-equilibrium gas content
bhtemp	Gsamvar.h	Bottomhole Temperature
bpslop	Gsamvar.h	Backpressure Exponent
bsncod	Gsamvar.h	Basin Code
CAOF	Type5.h	Calculated Absolute Open Flow on an Annual Basis
cap_base	Cashflow.h	Dep/Cap Base (tci-tciadj)
casename	Global.h	case name
casenm	Cost.h	case name
catcf	Cashflow.h	Cumulative After Tax Cash Flow
ce	Tax_nat.h	Capitalize Environmentals?
cgpr93	Gsamvar.h	Cumulative Gas Production to 1993
ChgTim	Type5.h	Year in which Automatic refrac or infill occurred
chlcon	Gsamvar.h	Cl Concentration of Produced Water
cidc	Tax_nat.h	Capitalize Intangible Drilling Cost?
cmpwat	Gsamvar.h	Compressibility of Water
CncCO2	Type1.h	CO2 fraction
CncH2S	Type1.h	H2S fraction
CncN2	Type1.h	N2 fraction
co2	Gsamvar.h	Carbon Dioxide Contamination
coi	Tax_nat.h	Capitalize Other Intangibles?

Variable Name	Location	Description
comp	Costing.h	compressor cost
comp_oam	Unitcost.h	compressor O&M
comp_vc	Cost.h	compressor O&M
comp_w	Unitcost.h	Compressor Cost
compfr	Gsamvar.h	Formation Compressibility
Cond	Type9.h	fracture conductivity
corr_yr	Global.h	correction year for model stabilization for history check
cost_bhp	Cost.h	Cost of Compressor Installation
county	Gsamvar.h	County Code
credamt	Tax_nat.h	Flag that allows AMT to be credited in future years
credit_npv	Npv.h	Credits NPV
CumGas	Type5.h	Cumulative Gas Produced Per Well
datcf	Cashflow.h	Discounted After Tax Cash Flow
datcf_2	Welldata.h	discounted atcf @2
datcf_5	Welldata.h	discounted atcf @5
dbtcf_2	Welldata.h	discounted btcf @2
dbtcf_5	Welldata.h	discounted btcf @5
dcstf	Cost.h	Drilling cost factor
Delta	Type5.h	Time Step
dep_crd	Cashflow.h	Depletion Credits
depcls	Gsamvar.h	Depositional Class
depggla_npv	Npv.h	Depletable G&G/LA NPV
deplet	Cashflow.h	DEPLETion
depr	Cashflow.h	Depreciation
depth	Gsamvar.h	Depth
Depth1	Type4.h	Depth
deptss	Gsamvar.h	Depth Subsea
dggla	Cashflow.h	Depletable G&G and lease acquisition
Diam	Type4.h	Inner diameter of tubing
diam_tech	Tech.h	inside diam of tubing
disc	Field.h	DISCount rate
discyr	Gsamvar.h	Date of Reservoir Discovery
disfld	Gsamvar.h	Date of Field Discovery
dismth	Gsamvar.h	Reservoir Discovery Method
domopr	Gsamvar.h	Dominant Operator as of 1993
dpidcs	Cashflow.h	Deduction Portion of IDC CostS
Dpsi	Type6.h	Calculated drop in real gas potential caused by production
DQ	Type5.h	Change in Producing Rate
drive	Gsamvar.h	Dominant Drive Type
drl_inv	Npv.h	Drilling Cost NPV
dwc	Costing.h	Development Well cost
dwc_npv	Npv.h	Devlopment Cost NPV
dwc_reg	Cost.h	GSAM supply region for development well cost calculation
dwc_tan	Cost.h	% DWC Tangible
dwc_w	Unitcost.h	DWC unit cost

Variable Name	Location	Description
dwck	Cost.h	Intercept in development drilling equation
dwcx	Cost.h	First coefficient in development drilling equation
dwcxx	Cost.h	Second coefficient in development drilling equation
dwcxxx	Cost.h	Third coefficient in development drilling equation
eccm	Cost.h	Environmental Capital Cost Mult.
eec	Tax_nat.h	Expense Environomental Costs?
eggla	Cashflow.h	Expensed G&G and Lease Acquisition cost
eiacod	Gsamvar.h	EIA Field Code
eicap	Costing.h	Environmental Intangible CAPital costs
eidca	Cashflow.h	Excess Intangible Drilling Cost Addback
eitc	Tax_nat.h	Environmental Intangible Tax Credit?
eitcr	Tax_nat.h	Environmental Intangible Tax Credit Rate
env_cap	Unitcost.h	Environmental Capital multiplier
env_cap_n	Unitcost.h	Environmental Capital Cost
env_cap_w	Unitcost.h	Environmental Capital Cost
env_ee	Cost.h	Env. existing Exp
env_ei	Cost.h	Env. existing Intan
env_et	Cost.h	Env. existing Tang
env_g	Cost.h	Env. O&M -gas
env_ne	Cost.h	Env. new exp
env_nf	Cost.h	Env. cost per foot drilling
env_ni	Cost.h	Env. new intang
env_nt	Cost.h	Env. new tang
env_oam_g	Unitcost.h	environmental O&M - Gas
env_oam_l	Unitcost.h	environmental O&M - Wells
env_oam_n	Unitcost.h	environmental O&M - Wells
env_oam_w	Unitcost.h	envrionmental O&M - Water
env_w	Cost.h	Env. O&M -water
envei	Unitcost.h	env unit cost intang
envet	Unitcost.h	env unit cost tang
envni	Unitcost.h	env unit cost intang
envnt	Unitcost.h	env unit cost tang
envscn	Tax_nat.h	flag for environmental scenario
eoam	Costing.h	Environomental Oper. And Main. Cost
eoca	Cashflow.h	Environmental Operating Cost Addback
eoctc	Tax_nat.h	Environmental Operating Cost Tax Credit?
eoctcr	Tax_nat.h	Environmental Operating Cost Tax Credit Rate
eortc	Field.h	EOR Tax Credit
eortca	Cashflow.h	EOR Tax Credit Addback
eortcr	Tax_nat.h	EOR Tax Credit Rate
etcap	Costing.h	Environmental Tangible CAPital costs
ettc	Tax_nat.h	Environmental Tangible Tax Credit?
ettcr	Tax_nat.h	Environmental Tangible Tax Credit Rate
ewc	Costing.h	exploratory Well cost
ewc_fac	Cost.h	EWC factor
ewc_npv	Npv.h	Exploratory Cost NPV
ewc_reg	Cost.h	GSAM supply region for exploration cost calculations

Variable Name	Location	Description
ewc_tan	Cost.h	% EWC Tangible
ewc_w	Unitcost.h	EWC unit cost
expggla_npv	Npv.h	Expensed G&G/LA NPV
fac_n	Cost.h	step to calculate facilities well cost
fac_tan(qtech)	Cost.h	% Facilities Tangible
fac_w	Unitcost.h	Facilities Cost
faci_k	Cost.h	Facilities Cost Constant factor
faci_max	Cost.h	Facilities Cost maximum level
faci_reg	Cost.h	facilities cost regional counter
faci_s	Cost.h	Facilities Cost slope factor
fedrate	Tax_nat.h	federal income tax rate used for GSAMID
fedrate_can	Tax_nat.h	Canada federal income tax rate
fedrate_us	Tax_nat.h	U.S. federal income tax rate
fedtax	Cashflow.h	FEDeral income TAXes
fedtaxc	Cashflow.h	FEDeral TAX Credits
fieldnm	Field.h	Field name
files	Global.h	variable to store *.GSM file name
fldnam	Gsamvar.h	Field name
fldtype	Gsamvar.h	Type of field
frac_fed	Gsamvar.h	Fraction of the reservoir on federal land
fraccn	Gsamvar.h	Induced Fracture Conductivity
fraccn_tech	Tech.h	frac conductivity
fracfl	Gsamvar.h	Fracture Flow Parameter
fracpo	Gsamvar.h	Induced Fracture Porosity
fracsk	Gsamvar.h	Induced Fracture Skin Factor
fracsk_tech	Tech.h	skin factors
fracsp	Gsamvar.h	Natural Fracture Spacing
fracwi	Gsamvar.h	Natural Fracture Width
fracxf	Gsamvar.h	Fracture Half Length or Length of Contact
fracxf_tech	Tech.h	frac half length
FrcSpc	Type3.h	Natural fracture spacing
fsclas	Gsamvar.h	Field size class
fsttax	Tax_reg.h	Forgiveness of STate TAXes?
fti	Cashflow.h	Federal Taxable Income
fxoam_k	Cost.h	fixed O&M Cost Constant subtor
fxoam_max	Cost.h	fixed O&M Cost maximum level
fxoam_n	Cost.h	number of steps
fxoam_reg	Cost.h	fixed O&M regional counter
fxoam_s	Cost.h	fixed O&M Cost slope subtor
fxoam_w	Unitcost.h	Fixed O&M
g_prd_npv	Npv.h	Gas Production NPV
ga_cap	Cashflow.h	G&A on Capitalized Items
ga_cap_m	Cost.h	G&A Capital Multiplier
ga_exp	Cashflow.h	G&A on Expensed Items
ga_exp_m	Cost.h	G&A Expense Multiplier
gas_sev	Tax_reg.h	Severance tax RATE (%)
gas_sev_p	Tax_reg.h	Severance tax RATE

Variable Name	Location	Description
gascon	Gsamvar.h	Initial Gas Concentration
GasCon1	Type8.h	Coal bed methane gas content
gasgrv	Gsamvar.h	Specific Gravity of Dry Gas
GasGrv1	Type1.h	Specific Gravity of Dry Gas
gasprd82	Gsamvar.h	gas production in 1982
gasprd83	Gsamvar.h	gas production in 1983
gasprd84	Gsamvar.h	gas production in 1984
gasprd85	Gsamvar.h	gas production in 1985
gasprd86	Gsamvar.h	gas production in 1986
gasprd87	Gsamvar.h	gas production in 1987
gasprd88	Gsamvar.h	gas production in 1988
gasprd89	Gsamvar.h	gas production in 1989
gasprd90	Gsamvar.h	gas production in 1990
gasprd91	Gsamvar.h	gas production in 1991
gasprd92	Gsamvar.h	gas production in 1992
gasprd93	Gsamvar.h	gas production in 1993
gasprod	Field.h	GAS PRODuction
gassat	Gsamvar.h	Initial Gas Saturation
gg	Costing.h	G&G costs
gg_fac	Cost.h	G&G factor
ggctc	Tax_nat.h	G&G Tangible Tax Credit (Depleted)
ggctcr	Tax_nat.h	G&G Tangible Tax Credit Rate (Depleted)
ggetc	Tax_nat.h	G&G Intangible Tax Credit (Expensed)
ggetcr	Tax_nat.h	G&G Intangible Tax Credit Rate (Expensed)
ggla	Cashflow.h	G&G/Lease Addback
gl_rat	Field.h	Gas/Liquid Ratio
gprice	Global.h	Gas PRICE
gravpen	Costing.h	GRAVity PENalty
gross_npv	Npv.h	Gross Sales NPV
grspay	Gsamvar.h	Gross Pay Thickness
grsv93	Gsamvar.h	Proved Gas Reserves End of 1993
gsamid	Gsamvar.h	Unique GSAM Identification Number
gsamsr	Gsamvar.h	GSAM supply region
gwr93	Gsamvar.h	1993 Gas-Water Ratio
h2odep	Gsamvar.h	Water Depth
h2ooam_w	Unitcost.h	surface O&M - Water
h2oprod	Field.h	WATER PRODuction
h2osat_fac	Geology.h	factor for water saturation
h2s	Gsamvar.h	Hydrogen Sulfide Contamination
HalfLn	Type9.h	fracture half length
heatvl	Gsamvar.h	Heating Value
HorLen	Type9.h	Horizontal well length
hurdle	Field.h	hurdle Rates (%)
icap	Costing.h	Intangible CAPital
idca	Cashflow.h	Intangible Drilling Cost Addback
idcpamt	Cashflow.h	IDC Preference for AMT
idctc	Tax_nat.h	Intangible Drilling Cost Tax Credit

Variable Name	Location	Description
idctcr	Tax_nat.h	Intangible Drilling Cost Tax Credit Rate
iea	Cashflow.h	Intangible Environmental Addback
ii	Cashflow.h	Intangible Investment
Iloc	Type8.h	location of the coal bed reservoir
Imod	Type10.h	module type
inj	Costing.h	INJectant Costs
inporf	Gsamvar.h	Interporosity Flow Factor
int_npv	Npv.h	Intangible Investment NPV
intadd	Cashflow.h	Total INTangible ADDback
intang_dwc	Cashflow.h	intangible development cost
intang_ewc	Cashflow.h	intangible exploratory cost
intang_m	Unitcost.h	Intangible multiplier (scalar)
intcap	Cashflow.h	INTangibles CAPitalized
ip	Field.h	independent producer
ipd	Tax_nat.h	Intangible drilling cost Preference Deduction
ipdr	Tax_nat.h	Independent Producer Depletion Rate
ira	Tax_nat.h	max alt. min. tax reduction for independents
istartappl	Gsamvar.h	Start year of production for Appalachian reservoirs
iuncloc	Gsamvar.h	Coal/Shale location (Eastern, Western, etc.)
iunctype	Gsamvar.h	Coal/Shale type (wet, dry, etc)
jlen_tech	Tech.h	Horizontal well length
Jtyp	Type9.h	variable to indicate horizontal/vertical well
jtyp_tech	Tech.h	well type by technology
KAqTyp	Type8.h	Aquifer type (marginal, infinite acting, etc.)
Kshut	Type8.h	Variable used to shuting well
KUnCon	Type8.h	Coal/Shale type (wet, dry, etc)
kwinyr	Type_out.h	Window year
la	Costing.h	Lease Acquisition costs
lactc	Tax_nat.h	Lease Acq. Tangible Tax Credit (Depleted)
lactcr	Tax_nat.h	Lease Acq. Tangible Tax Credit Rate (Depleted)
laetc	Tax_nat.h	Lease Acq. Intangible Tax Credit (Expensed)
laetcr	Tax_nat.h	Lease Acq. Intangible Tax Credit Rate (Expensed)
langpr	Gsamvar.h	Langmuir Pressure
langvl	Gsamvar.h	Langmuir Volume
lastyr	Cashflow.h	Last year of operation
lat	Gsamvar.h	Latitude of Reservoir Centroid
lbc_fac	Cost.h	lease bonus cost factor
lbc_frac	Unitcost.h	lease bonus cost factor
lon	Gsamvar.h	Longitude of Reservoir Centroid
masp	Welldata.h	Mininum Acceptable Supply Price
module	Gsamvar.h	Type curve module identifier
n2	Gsamvar.h	Nitrogen Contamination
Narray	Type2.h	number of data elements in an array
ndwcreg	Cost.h	number of regions specified for development drilling cost specification
netpay	Gsamvar.h	Total Net Pay in Designated Formation
netpay_fac	Geology.h	factor for net pay

Variable Name	Location	Description
netsales	Cashflow.h	NET SALES
newcreg	Cost.h	number of regions specified for incremental environmental cost specification
nglfact	Gsamvar.h	Barrels NGL/Mmcf dry gas
nglprd82	Gsamvar.h	natural gas liquids production in 1982
nglprd83	Gsamvar.h	natural gas liquids production in 1983
nglprd84	Gsamvar.h	natural gas liquids production in 1984
nglprd85	Gsamvar.h	natural gas liquids production in 1985
nglprd86	Gsamvar.h	natural gas liquids production in 1986
nglprd87	Gsamvar.h	natural gas liquids production in 1987
nglprd88	Gsamvar.h	natural gas liquids production in 1988
nglprd89	Gsamvar.h	natural gas liquids production in 1989
nglprd90	Gsamvar.h	natural gas liquids production in 1990
nglprd91	Gsamvar.h	natural gas liquids production in 1991
nglprd92	Gsamvar.h	natural gas liquids production in 1992
nglprd93	Gsamvar.h	natural gas liquids production in 1993
niat	Cashflow.h	Net Income After Taxes
nibt	Cashflow.h	Net Income Before Taxes
nibta	Cashflow.h	Net Income Before Tax Addback
nifoag	Cashflow.h	Net Income From Oil And Gas
nil	Tax_nat.h	Net Income Limitations?
nilb	Cashflow.h	Net Income Limitation Base
nill	Tax_nat.h	Net Income Limitation Limit
npv	Npv.h	NPV
npv_prd	Welldata.h	npv value of production
nreg	Global.h	number of regions
nreg_faci	Cost.h	number of regions with facilities well cost
nreg_fx	Cost.h	number of regions with fixed o&m
nrestype	Geology.h	number of reservoir types
ntax_st	Tax_reg.h	number of tax regions
ntech	Cost.h	number of technologies
ntech_st	Tech.h	number of states for which proration is specified
nwell	Field.h	Number of wells
nyr	Global.h	number of years
nyrset	Global.h	number of economic years
o_prd_npv	Npv.h	Oil Production NPV
oam	Costing.h	O&M
oam_gas	Cost.h	Gas O&M
oam_h2o	Cost.h	Surface O&M H2O
oam_inc	Cost.h	O&M - Incremental per 1000 feet
oam_m	Unitcost.h	O&M Multiplier (scalar)
ogip	Gsamvar.h	Reservoir Volumetric Original Gas in Place
OGIP1	Type3.h	Original gas in place
oia	Cashflow.h	Other Intangible Addbacks
oil_sev	Tax_reg.h	Severance tax RATE (%)
oil_sev_p	Tax_reg.h	Severance tax RATE ($/B)
oilprd82	Gsamvar.h	oil production in 1982

Variable Name	Location	Description
oilprd83	Gsamvar.h	oil production in 1983
oilprd84	Gsamvar.h	oil production in 1984
oilprd85	Gsamvar.h	oil production in 1985
oilprd86	Gsamvar.h	oil production in 1986
oilprd87	Gsamvar.h	oil production in 1987
oilprd88	Gsamvar.h	oil production in 1988
oilprd89	Gsamvar.h	oil production in 1989
oilprd90	Gsamvar.h	oil production in 1990
oilprd91	Gsamvar.h	oil production in 1991
oilprd92	Gsamvar.h	oil production in 1992
oilprd93	Gsamvar.h	oil production in 1993
oilprod	Field.h	OIL PRODuction
oitc	Tax_nat.h	Other Intangible Tax Credit?
oitcr	Tax_nat.h	Other Intangible Tax Credit Rate
onoffs	Gsamvar.h	Onshore/Offshore (1=Onshore, 2=State Offshore, 3=Federal Offshore)
oprice	Global.h	Oil PRICE
otc	Costing.h	Other Tangible Capital
pay_tech	Tech.h	pay continuity factor
paydsp	Gsamvar.h	Pay Dispersion Function
pdr	Tax_nat.h	Percent Depletion Rate (%)
pdry_dev	Cost.h	Percent dry hole cost as & of Dev.
peakrate	Field.h	peak production rate
perhor	Gsamvar.h	Effective Horizontal Permeability
Perm	Type3.h	Effective Horizontal Permeability
perm_fac	Geology.h	factor for permeability
PermMa	Type3.h	Matrix Permeability
permtx	Gsamvar.h	Matrix Permeability
pervrt	Gsamvar.h	Effective Vertical Permeability
pggc	Tax_reg.h	Percent of G&G as Tangible (depletable)
piic	Tax_nat.h	Percent of Intan. Inv. to Capitalize
Pinit	Type3.h	Initial Reservoir Pressure
PL	Type8.h	Langmuir Pressure
plac	Tax_reg.h	Percent Lease Acquisition cost Tangible
plycod	Gsamvar.h	4 digit play code
Pmin	Type5.h	Mininum Wellhead Pressure
por_fac	Geology.h	factor for porosity for pay grade distribution
porcur	Gsamvar.h	Current Total Effective Porosity
PorMa	Type3.h	Matrix Porosity
pormtx	Gsamvar.h	Matrix Porosity
Poros	Type3.h	Porosity
portot	Gsamvar.h	Total Effective Initial Porosity
Ppc	Type1.h	Pseudo Critical Pressure
Prbh	Type5.h	Bottomhole Pressure
prdwel82	Gsamvar.h	Producing Wells 1982
prdwel83	Gsamvar.h	Producing Wells 1983
prdwel84	Gsamvar.h	Producing Wells 1984

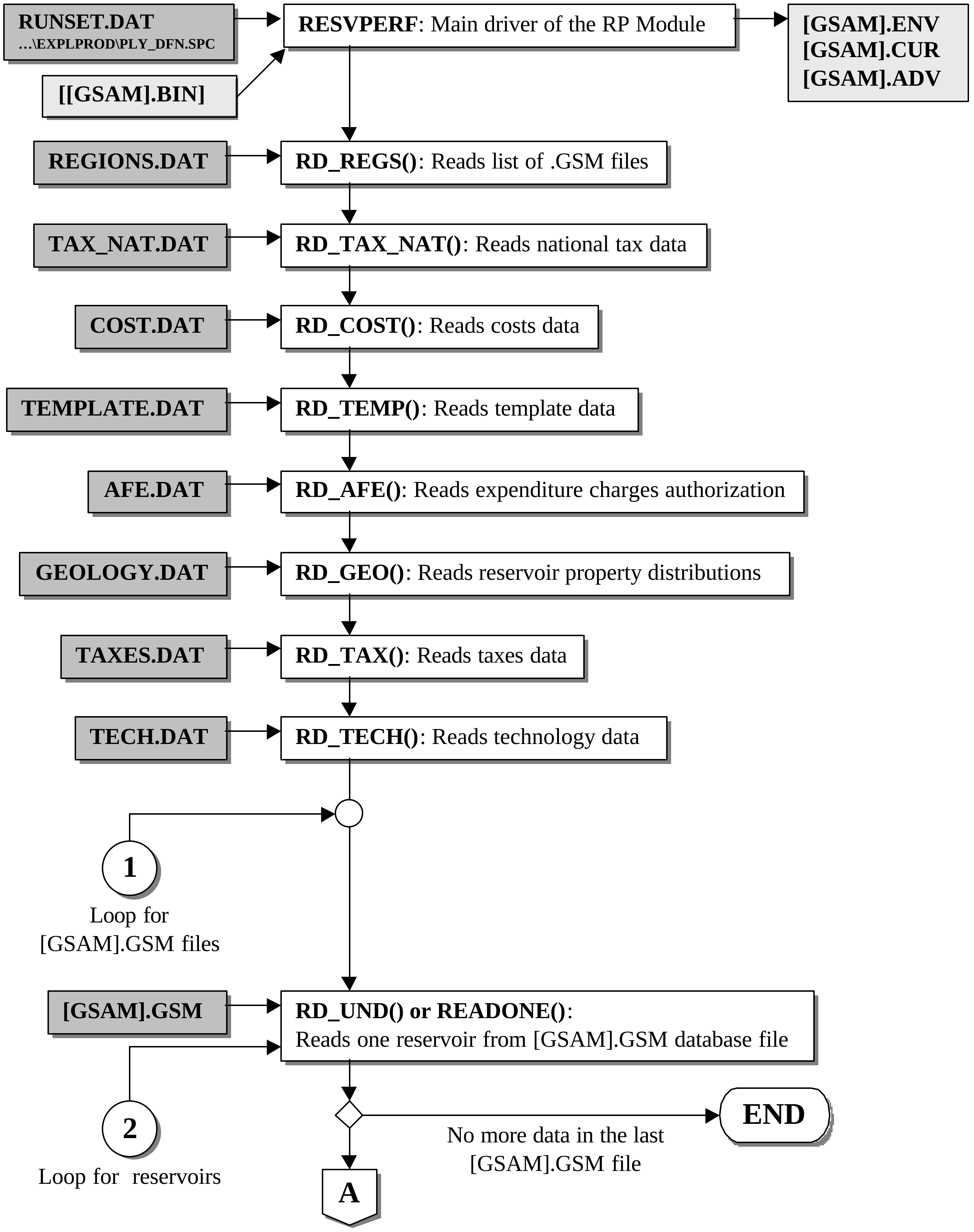
Variable Name	Location	Description
prdwel85	Gsamvar.h	Producing Wells 1985
prdwel86	Gsamvar.h	Producing Wells 1986
prdwel87	Gsamvar.h	Producing Wells 1987
prdwel88	Gsamvar.h	Producing Wells 1988
prdwel89	Gsamvar.h	Producing Wells 1989
prdwel90	Gsamvar.h	Producing Wells 1990
prdwel91	Gsamvar.h	Producing Wells 1991
prdwel92	Gsamvar.h	Producing Wells 1992
prdwel93	Gsamvar.h	Producing Wells 1993
PreAry	Type2.h	Pressure Array
PreAvg	Type5.h	Average Reservoir Pressure
Premin	Type7.h	Mininum Allowable Wellhead Pressure
presin	Gsamvar.h	Initial Reservoir Shut-In Pressure (PSIA)
prob_dry	Cost.h	Probability of Dry hole for development drilling
prorat	Gsamvar.h	Proration - Rule for Wells/Reservoir
prorat_tech	Tech.h	proration factor by technology
proration	Tech.h	proration factor
provcod	Gsamvar.h	Province code
prscur	Gsamvar.h	Current Bottomhole Shutin Pressure
prsdsp	Gsamvar.h	Desorption Pressure
prsflw	Gsamvar.h	Current Bottomhole Flowing Pressure
prssys	Gsamvar.h	Operating System Back Pressure (PSIA)
Prwh	Type5.h	Wellhead Pressure
PsiAry	Type2.h	Real gas potential array
PsiCon	Type6.h	variable to convert dimensionless pressure to real gas potential
psys_tech	Tech.h	min system pressure
pzslop	Gsamvar.h	Slope of Cumulative Production vs. p/z
qafe	Dimen.h	number of afe categories
qcase	Dimen.h	number of cases (in type curve
qfield	Dimen.h	max number of fields per region
Qg	Type5.h	Gas Production Rate per Well
qline	Dimen.h	number of lines in type curve file
Qmax	Type5.h	Maximum Flow Rate
qnpv	Dimen.h	number of NPV calculations
qpay	Dimen.h	number of paygrades
qplay	Dimen.h	Total number of plays that could be read from ply_dfn.spc file
qreg	Dimen.h	max number of regions
qrestype	Dimen.h	number of reservoir types
qrgst	Dimen.h	maximum number of regions allowed
qstate	Dimen.h	number of states
qstep	Dimen.h	number of step in cost function
qtech	Dimen.h	max number of technologies
Qw	Type8.h	water flow rate by pay grade
QwMax	Type8.h	maximum water flow rate
Qwtr	Type8.h	Total water flow rate for reservoir

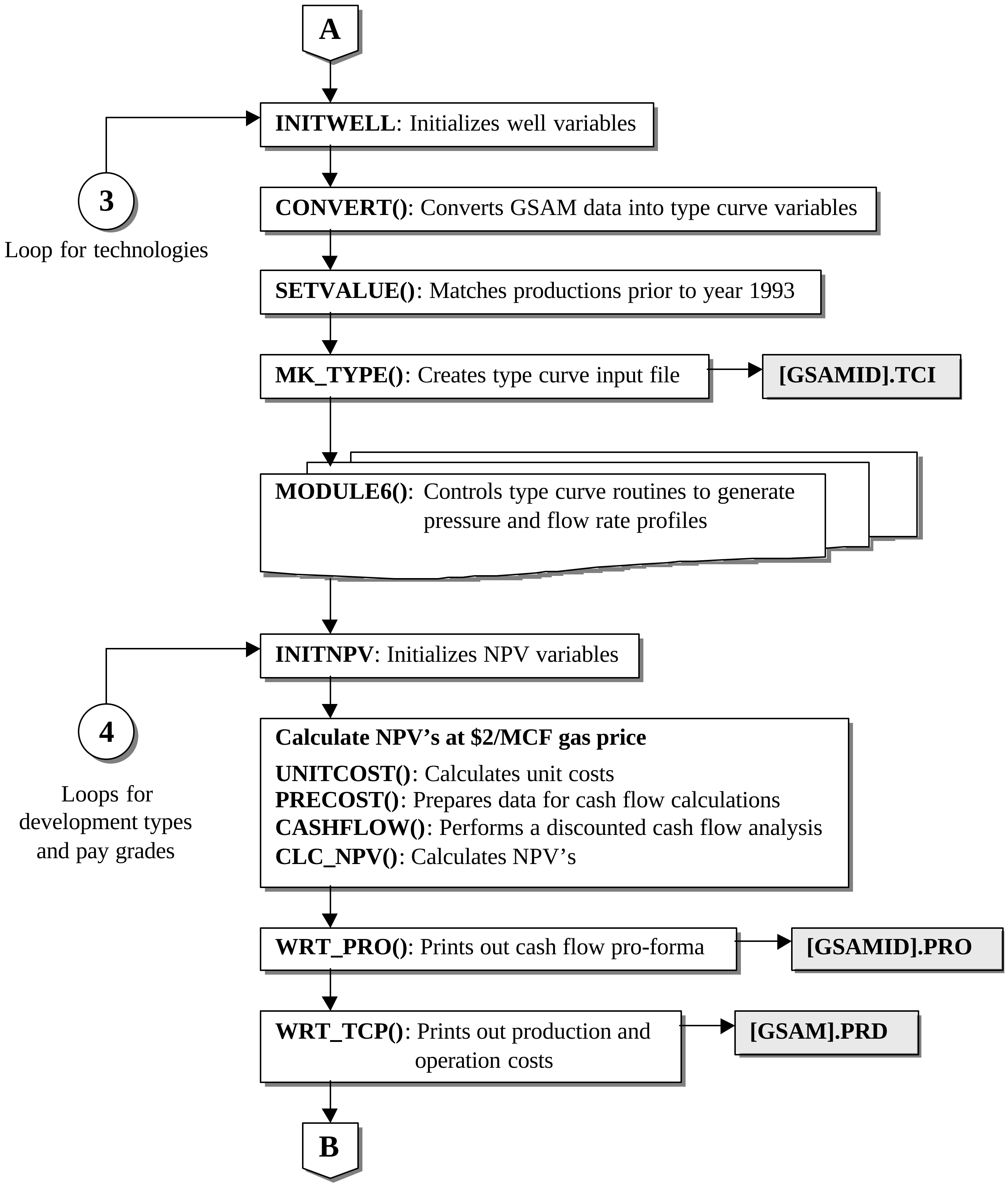
Variable Name	Location	Description
qyr	Dimen.h	number of years in time horizon
range	Gsamvar.h	Range name
ratmax	Type7.h	Maximum Rate From Field
recomp	Costing.h	recompletion cost
regnm	Global.h	Region name
res_map	Geology.h	reservoir type map
rescod	Gsamvar.h	Reservoir Code
restype	Gsamvar.h	Lithology Type
RhoMa	Type8.h	Coal density
royrate	Tax_reg.h	royalty rate (%)
rsvcls	Gsamvar.h	AAPG Reservoir Size Class
rsvnam	Gsamvar.h	Reservoir Formation Name
runtype	Global.h	Counter flag "Yes" or "NO" for type curve run
Rw	Type4.h	Well Radius
Salin	Type3.h	Water salinity, ppm by weight
sevtax	Cashflow.h	Severance tax
sfit	Cashflow.h	Selected Federal Income Taxes
sgdvyr	Gsamvar.h	Year Significant Development Drilling Starts
SgTrap	Type8.h	Trapped gas saturation
shutwel82	Gsamvar.h	Shutin Wells 1982
shutwel83	Gsamvar.h	Shutin Wells 1983
shutwel84	Gsamvar.h	Shutin Wells 1984
shutwel85	Gsamvar.h	Shutin Wells 1985
shutwel86	Gsamvar.h	Shutin Wells 1986
shutwel87	Gsamvar.h	Shutin Wells 1987
shutwel88	Gsamvar.h	Shutin Wells 1988
shutwel89	Gsamvar.h	Shutin Wells 1989
shutwel90	Gsamvar.h	Shutin Wells 1990
shutwel91	Gsamvar.h	Shutin Wells 1991
shutwel92	Gsamvar.h	Shutin Wells 1992
shutwel93	Gsamvar.h	Shutin Wells 1993
Skin	Type5.h	Skin factor
slope1	Welldata.h	Drilling slope
slope2	Welldata.h	non-drilling slope
smar	Tax_nat.h	Six Month Amortization Rate (%)
solgas	Gsamvar.h	Gas Solubility in Brine
srptim	Gsamvar.h	Pseudo Steady State Desorption Time
state	Gsamvar.h	State Code
statin	Gsamvar.h	Initial Development Status
stim	Costing.h	stimulation cost
stim_w	Unitcost.h	Stimulation Cost
stimfac	Cost.h	Stimulation Cost, fraction
strate	Tax_reg.h	State income tax RATE (%)
sttax	Cashflow.h	STate Income TAXes
Swi	Type3.h	initial water saturation
tan_npv	Npv.h	Tangible Investment NPV
tang_dwc	Cashflow.h	Tangible development cost

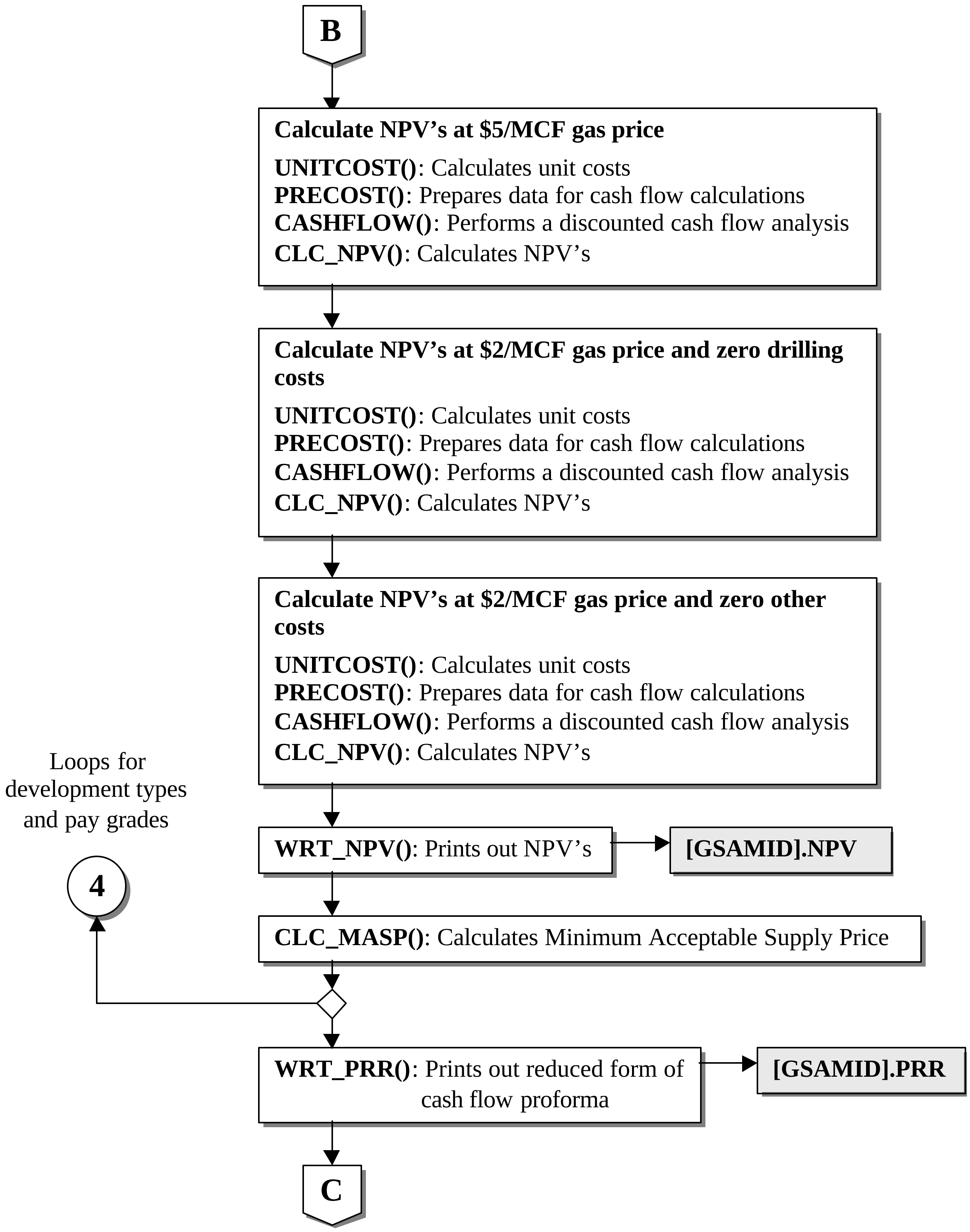
Variable Name	Location	Description
tang_ewc	Cashflow.h	Tangible exploratory cost
tang_m	Unitcost.h	Tangible Multiplier (scalar)
tax_npv	Npv.h	Tax NPV
tax_st	Tax_reg.h	region identifier
tci	Cashflow.h	Total Capitalized Investments
tciadj	Cashflow.h	Total Capitalized Investments ADJustment
tcoii	Tax_nat.h	Tax Credit On Intangible Investments
tcoti	Tax_nat.h	Tax Credit On Tangible Investments
Tdes	Type8.h	Desorption Time
tdtc	Tax_nat.h	Tangible Development Tax Credit?
tdtcr	Tax_nat.h	Tangible Development Tax Credit Rate
tech_st	Tech.h	state specifications for proration specification
technm	Tech.h	Name of technology
tfit	Cashflow.h	Tenative Federal Income Taxes
Thick	Type3.h	Reservoir Thickness
ti	Cashflow.h	Tangible Investments
timchg	Type7.h	Year in which automatic refrac or infill occurred
TimCon	Type6.h	variable converts dimensionless time to real time
Time	Type5.h	Time in years
toc	Cashflow.h	Total Operating Cost
toc_npv	Npv.h	Total Operating Cost NPV
tot_cap_2	Welldata.h	total capital @$2/Mcf
tot_cap_5	Welldata.h	total capital @$5/Mcf
tot_inv	Npv.h	total investments NPV
totalcst	Npv.h	total cost of each scenario
totalgas	Field.h	Total GAS Production
totaloil	Field.h	Total Oil Production
totoam	Costing.h	Total O&M cost
totwel82	Gsamvar.h	Total Wells 1982
totwel83	Gsamvar.h	Total Wells 1983
totwel84	Gsamvar.h	Total Wells 1984
totwel85	Gsamvar.h	Total Wells 1985
totwel86	Gsamvar.h	Total Wells 1986
totwel87	Gsamvar.h	Total Wells 1987
totwel88	Gsamvar.h	Total Wells 1988
totwel89	Gsamvar.h	Total Wells 1989
totwel90	Gsamvar.h	Total Wells 1990
totwel91	Gsamvar.h	Total Wells 1991
totwel92	Gsamvar.h	Total Wells 1992
totwel93	Gsamvar.h	Total Wells 1993
Tpc	Type1.h	Pseudo Critical Temperature
transcst	Costing.h	TRANSportation CoST
trapty	Gsamvar.h	Trap Type
twlspac	Gsamvar.h	Target Well Spacing
twnshp	Gsamvar.h	Township name
type_gas	Type_out.h	type curve gas production
type_ibhp	Type_out.h	infill bottom hole pressure

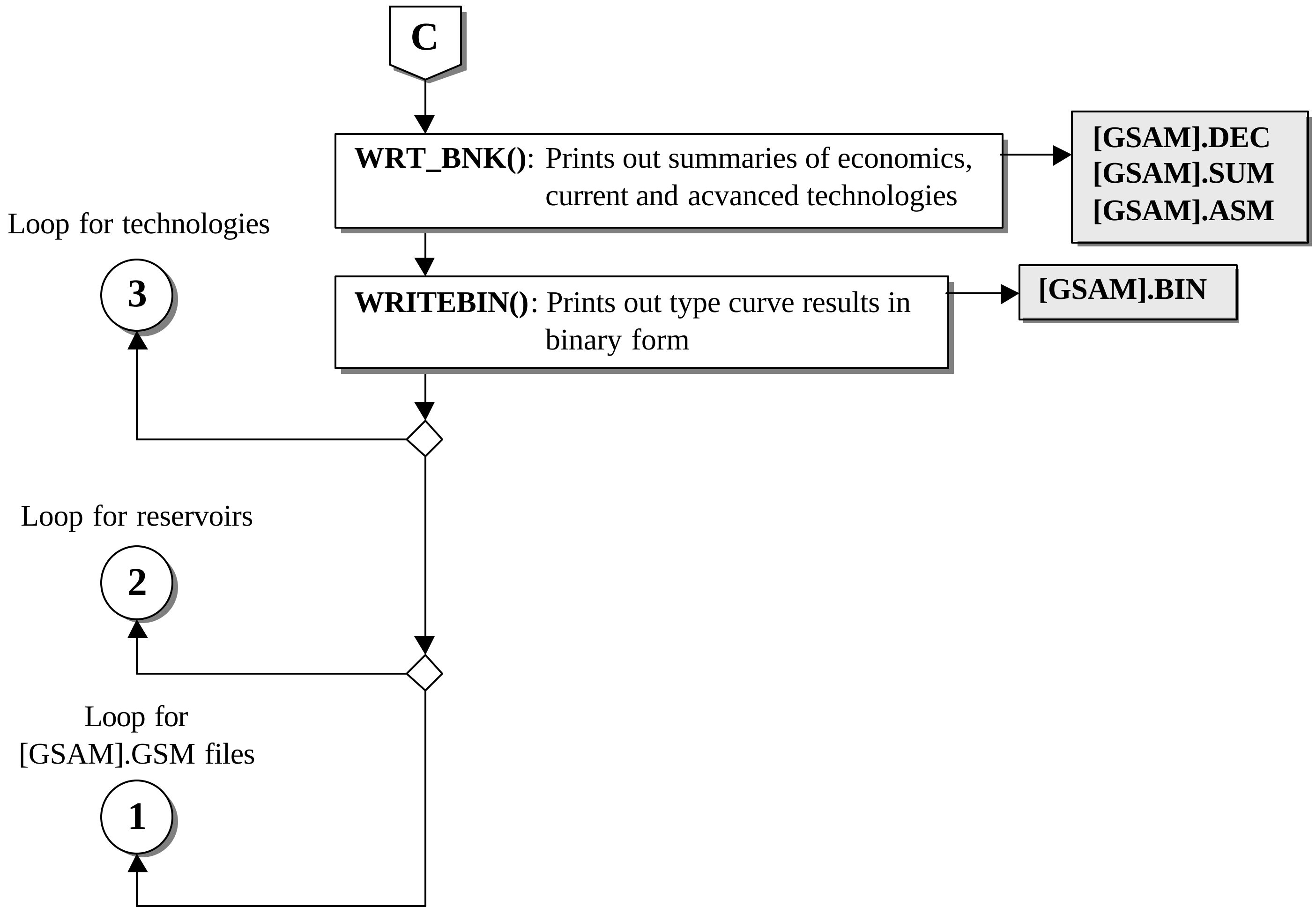
Variable Name	Location	Description
type_ogip	Type_out.h	Original Gas in place
type_pbhp	Type_out.h	prim. bottom hole pressure
type_pwhp	Type_out.h	prim. well head pressure
type_well	Type_out.h	number of wells
uamti	Cashflow.h	Unadjusted AMT Income
ucpamt	Cashflow.h	Useable Credits for Past AMT
udatcf_2	Welldata.h	undiscounted atcf @ $2/MCF
udatcf_5	Welldata.h	undiscounted atcf @ $5/MCF
udbtcf_2	Welldata.h	undiscounted btcf @ $2/MCF
udbtcf_5	Welldata.h	undiscounted btcf @ $5/MCF
Va	Type1.h	Viscosity
VisAry	Type2.h	Viscosity Array for Interpolation
VL	Type8.h	Langmuir Volume
voam_g	Unitcost.h	surface O&M - Gas
watsab	Gsamvar.h	Abandonment Water Saturation
watsac	Gsamvar.h	Current Water Saturation
watsaf	Gsamvar.h	Fracture Water Saturation
watsat	Gsamvar.h	Initial Water Saturation
wdtim_tech	Tech.h	time to drill infill well in water drive reservoir
We	Type8.h	Water influx in cubic feet
WeD	Type8.h	Net water influx
weldrn	Gsamvar.h	Well Drainage Area
welrad	Gsamvar.h	Wellbore Radius
WePrev	Type8.h	Water influx in previous time step
win_yr	Welldata.h	year at which minimum press not met
wlspac	Gsamvar.h	Well Spacing
wlspac1	Gsamvar.h	Well Spacing
Wp	Type8.h	Total water production in time step
wrad_tech	Tech.h	well radius by technology
Wspace	Type4.h	Well spacing by pay grade
WtrInf	Type8.h	Water influx
yr1	Tax_nat.h	number of years for tcoti calculations
yr2	Tax_nat.h	number of years for tcoii calculations
yr3	Tax_reg.h	number of years for fsttax calculations

MAIN PROGRAM RESVPERF


RUNSET.DAT
…\EXPLPROD\PLY_DFN.SPC
[[GSAM].BIN]
RESVPERF: Main driver of the RP Module
[GSAM].ENV
[GSAM].CUR
[GSAM].ADV
REGIONS.DAT
RD_REGS(): Reads list of .GSM files
TAX_NAT.DAT
RD_TAX_NAT(): Reads national tax data
COST.DAT
RD_COST(): Reads costs data
TEMPLATE.DAT
RD_TEMP(): Reads template data
AFE.DAT
RD_AFE(): Reads expenditure charges authorization
GEOLOGY.DAT
RD_GEO(): Reads reservoir property distributions
TAXES.DAT
RD_TAX(): Reads taxes data
TECH.DAT
RD_TECH(): Reads technology data
1
Loop for [GSAM].GSM files
[GSAM].GSM
RD_UND() or READONE(): Reads one reservoir from [GSAM].GSM database file
2
Loop for reservoirs
No more data in the last [GSAM].GSM file
END
A


A
INITWELL: Initializes well variables
3
Loop for technologies
CONVERT(): Converts GSAM data into type curve variables
SETVALUE(): Matches productions prior to year 1993
MK_TYPE(): Creates type curve input file
[GSAMID].TCI
MODULE6(): Controls type curve routines to generate pressure and flow rate profiles
INITNPV: Initializes NPV variables
4
Loops for development types and pay grades
Calculate NPV's at $2/MCF gas price
UNITCOST(): Calculates unit costs
PRECOST(): Prepares data for cash flow calculations
CASHFLOW(): Performs a discounted cash flow analysis
CLC_NPV(): Calculates NPV's
WRT_PRO(): Prints out cash flow pro-forma
[GSAMID].PRO
WRT_TCP(): Prints out production and operation costs
[GSAM].PRD
B

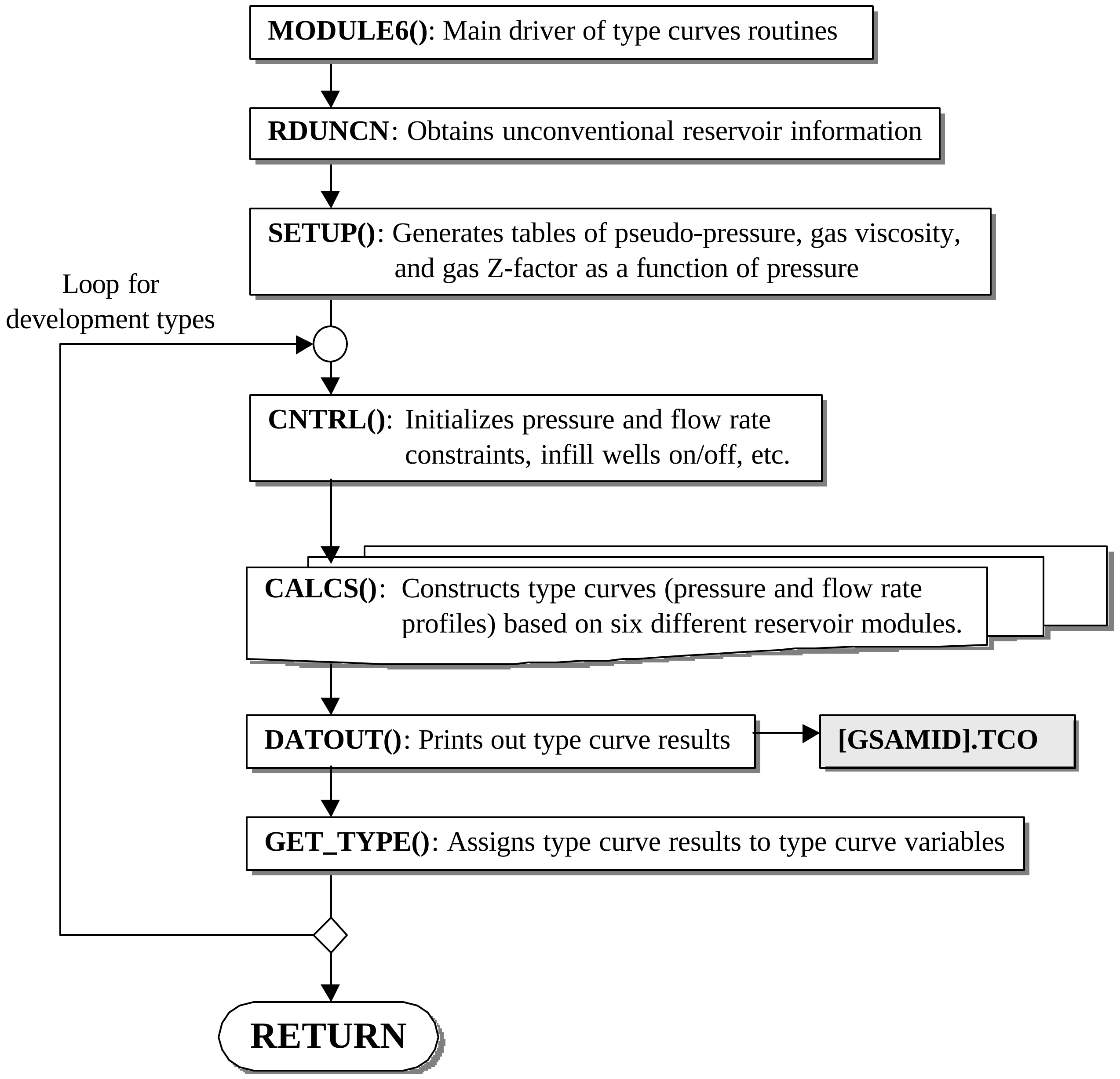

B
Calculate NPV’s at $5/MCF gas price
UNITCOST(): Calculates unit costs
PRECOST(): Prepares data for cash flow calculations
CASHFLOW(): Performs a discounted cash flow analysis
CLC_NPV(): Calculates NPV’s
Calculate NPV’s at $2/MCF gas price and zero drilling costs
UNITCOST(): Calculates unit costs
PRECOST(): Prepares data for cash flow calculations
CASHFLOW(): Performs a discounted cash flow analysis
CLC_NPV(): Calculates NPV’s
Calculate NPV’s at $2/MCF gas price and zero other costs
UNITCOST(): Calculates unit costs
PRECOST(): Prepares data for cash flow calculations
CASHFLOW(): Performs a discounted cash flow analysis
CLC_NPV(): Calculates NPV’s
Loops for development types and pay grades
WRT_NPV(): Prints out NPV’s
[GSAMID].NPV
4
CLC_MASP(): Calculates Minimum Acceptable Supply Price
WRT_PRR(): Prints out reduced form of cash flow proforma
[GSAMID].PRR
C


C
WRT_BNK(): Prints out summaries of economics, current and acvanced technologies
[GSAM].DEC
[GSAM].SUM
[GSAM].ASM
WRITEBIN(): Prints out type curve results in binary form
[GSAM].BIN
Loop for technologies
3
Loop for reservoirs
2
Loop for [GSAM].GSM files
1

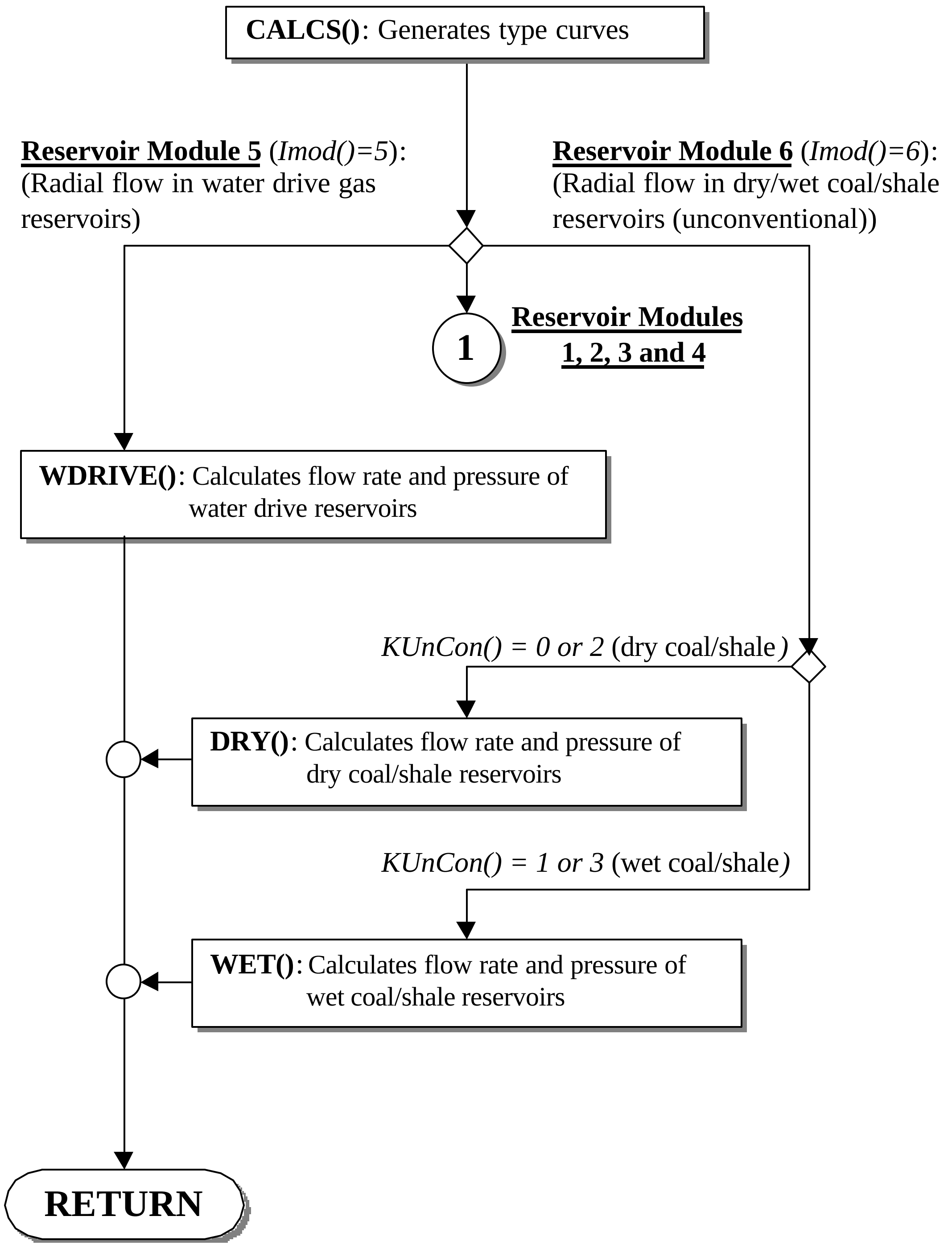
Legends

▨	Optional input/output files read/created
▨	Primary input files read
▨	Primary output files created
☐	Program files

SUB-PROGRAM MODULE6()


MODULE6(): Main driver of type curves routines
RDUNCN: Obtains unconventional reservoir information
SETUP(): Generates tables of pseudo-pressure, gas viscosity, and gas Z-factor as a function of pressure
Loop for development types
CNTRL(): Initializes pressure and flow rate constraints, infill wells on/off, etc.
CALCS(): Constructs type curves (pressure and flow rate profiles) based on six different reservoir modules.
DATOUT(): Prints out type curve results
[GSAMID].TCO
GET_TYPE(): Assigns type curve results to type curve variables
RETURN

SUB-PROGRAM CALCS()

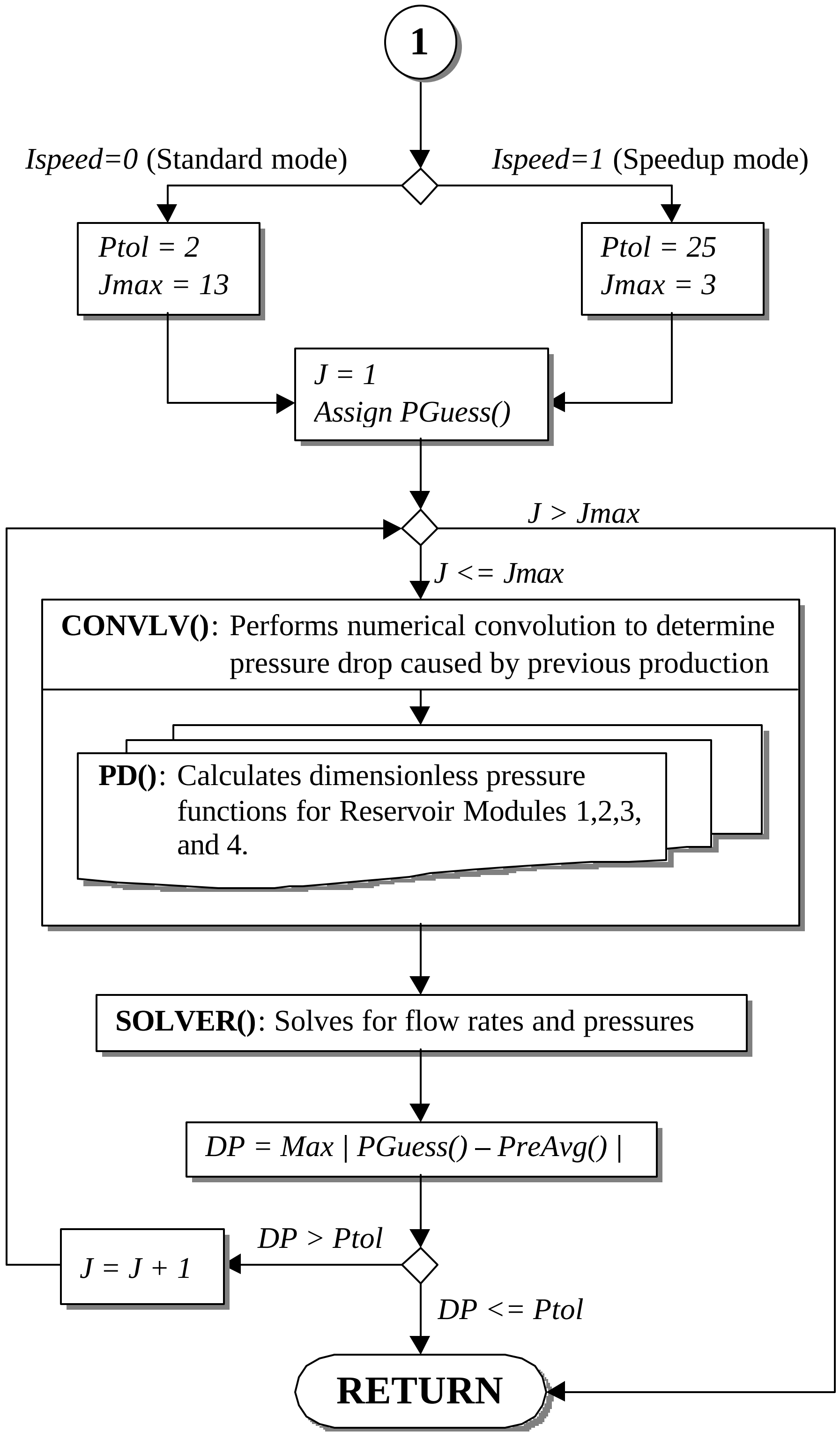


CALCS(): Generates type curves
Reservoir Module 5 (*Imod()=5*): (Radial flow in water drive gas reservoirs)
Reservoir Module 6 (*Imod()=6*): (Radial flow in dry/wet coal/shale reservoirs (unconventional))
1
Reservoir Modules 1, 2, 3 and 4
WDRIVE(): Calculates flow rate and pressure of water drive reservoirs
KUnCon() = 0 or 2 (dry coal/shale)
DRY(): Calculates flow rate and pressure of dry coal/shale reservoirs
KUnCon() = 1 or 3 (wet coal/shale)
WET(): Calculates flow rate and pressure of wet coal/shale reservoirs
RETURN

NOTES:

Imod()=reservoir module flag

KUnCon()=unconventional reservoir flag (0=dry coal, 1=wet coal, 2=dry shale, 3=wet shale)


1
Ispeed=0 (Standard mode)
Ispeed=1 (Speedup mode)
Ptol = 2
Jmax = 13
Ptol = 25
Jmax = 3
J = 1
Assign PGuess()
J > Jmax
J <= Jmax
CONVLV(): Performs numerical convolution to determine pressure drop caused by previous production
PD(): Calculates dimensionless pressure functions for Reservoir Modules 1,2,3, and 4.
SOLVER(): Solves for flow rates and pressures
DP = Max | PGuess() – PreAvg() |
DP > Ptol
J = J + 1
DP <= Ptol
RETURN

NOTES:

Ispeed=speedup flag (0=standard, 1=speedup)

Ptol=pressure tolerance (psi)

Jmax=maximum number of iteration

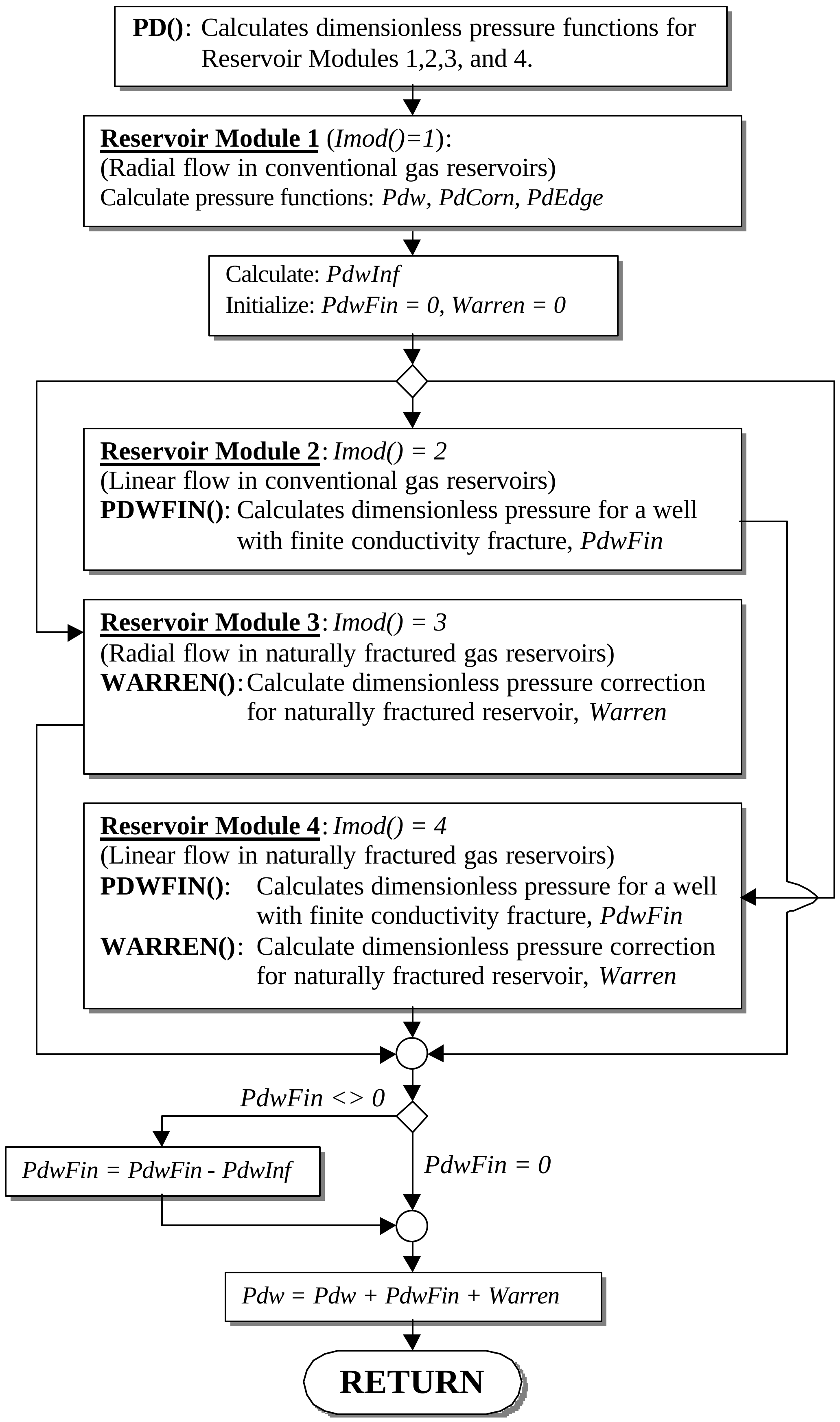
J=iteration counter

PGuess()=array of guessed pressures (psia)

PreAvg()=array of calculated average pressures (psia)

DP=Maximum pressure deviation (psi)

SUB-PROGRAM PD()


PD(): Calculates dimensionless pressure functions for Reservoir Modules 1,2,3, and 4.
Reservoir Module 1 (*Imod()=1*):
(Radial flow in conventional gas reservoirs)
Calculate pressure functions: *Pdw*, *PdCorn*, *PdEdge*
Calculate: *PdwInf*
Initialize: *PdwFin = 0, Warren = 0*
Reservoir Module 2: *Imod() = 2*
(Linear flow in conventional gas reservoirs)
PDWFIN(): Calculates dimensionless pressure for a well with finite conductivity fracture, *PdwFin*
Reservoir Module 3: *Imod() = 3*
(Radial flow in naturally fractured gas reservoirs)
WARREN(): Calculate dimensionless pressure correction for naturally fractured reservoir, *Warren*
Reservoir Module 4: *Imod() = 4*
(Linear flow in naturally fractured gas reservoirs)
PDWFIN(): Calculates dimensionless pressure for a well with finite conductivity fracture, *PdwFin*
WARREN(): Calculate dimensionless pressure correction for naturally fractured reservoir, *Warren*
PdwFin <> 0
PdwFin = PdwFin - PdwInf
PdwFin = 0
Pdw = Pdw + PdwFin + Warren
RETURN

NOTES:

Imod()=reservoir module flag

Pdw=dimensionless pressure at the well

PdCorn=dimensionless pressure at the corner of reservoir

PdEdge=dimensionless pressure at the second infill location

PdwInf=dimensionless pressure of a well in infinite reservoir

PdwFin=dimensionless pressure of a well with finite conductivity fracture in infinite reservoir

Warren=dimensionless pressure correction for naturally fractured reservoir

MAIN-PROGRAM RESVPERF

LOCATION: RESVPERF.FOR

MAIN THEME: This program serves as the main driver that controls the main flow of the GSAM Reservoir Performance (RP) Module.

CALLS: **Reading Routines**

RD_REGS() (in file GSAM_A.FOR)
Reads REGIONS.DAT file which contains information about the list of the .GSM files to be run through the RP Module and several YES/NO switches as indicators for opening specific files for consistency checks.

RD_TAX_NAT() (in file READINP.FOR)
Reads TAX_NAT.DAT which contains information about generic tax structure (capitalize versus expense switches) assumptions.

RD_COST() (in file READINP.FOR)
Reads COST.DAT which contains cost related information.

RD_TEMP() (in file GSAM_A.FOR)
Reads TEMPLATE.DAT (a template file used to generate type curve input parameters) which contains information on fluid and reservoir properties data, well data, field development, drive mechanism, and other type curve related data.

RD_AFE() (in file READINP.FOR)
Reads input file AFE.DAT which contains information on authorization for expenditure charges for a producer (not currently used)

RD_GEO() (in file READINP.FOR)
Reads GEOLOGY.DAT which contains information on reservoir property distributions by pay grade.

RD_TAX() (in file READINP.FOR)
Reads TAXES.DAT file which contains information on state income taxes, oil and gas severance taxes, and ad-voleram taxes.

RD_TECH() (in file READINP.FOR)
Reads TECH.DAT which contains information on number of technologies and data specifications for each technology.

RD_UND() (in file READONE.FOR)
Reads one record from the one-line format .GSM file.

READONE() (in file READONE.FOR)
Reads one record from the full database format .GSM file.

Initialization Routines

INIT_WELL (in file GSAM_B.FOR)
Initializes type curve and economic variables.

INITNPV (in file INITIAL.FOR)
Initializes NPV related variables.

Data Setup Routines

CONVERT() (in file CONVERT.FOR)
Converts the .GSM data into type curve variable names and distributes them on a pay grade level.

SETVALUE() (in file SETVALUE.FOR)
Predicts the production years of the reservoir prior to the year 1993 (used for history check for existing reservoirs).

Type Curve Routines

MK_TYPE() (in file MK_TYPE.FOR)
Creates an input file for the type curve module (MODULE6()).

MODULE6() (in file MODULE6A.FOR)
Controls the type curve modules in generating type curve data.

Costing Routines

UNITCOST() (in file UNITCOST.FOR)
Calculates per unit costs in $/MCF, $/Well and/or $/BBL.

PRECOST() (in file PRECOST.FOR)
Utilizes the unit cost data to create the cost streams to be fed to the cash flow routine CASHFLOW().

CASHFLOW() (in file CASHFLOW.FOR)
Performs a discounted cash flow analysis (i.e. performs a pro-forma cash flow analysis for every reservoir processed).

CLC_NPV() (in file WRT_PRO.FOR)

Performs Net Present Value (NPV) calculations at different price and cost assumptions.

Writing Routines

WRT_PRO() (in file WRT_PRO.FOR)
Writes out cash flow pro-forma to output file .PRO.

WRT_TCP() (in file WRT_PRO.FOR)
Writes out production and operating costs to output file .PRD, file fed to E&P Module.

WRT_NPV() (in file WRT_PRO.FOR)
Writes out NPV's to output file .NPV.

WRT_PRR() (in file GSAM_A.FOR)
Produces a reduced form of cash flow pro-forma in output file .PRR.

WRT_BNK() (in file GSAM_B.FOR)
Reports reserves, OGIP, etc. and summary of economics to output file .DEC. Also reports summary of current technology to output file .SUM and summary of advanced technology to output file .ASM.

WRITEBIN() (in file READONE.FOR)
Writes out type curve outputs to output file .BIN.

Miscellaneous Routines

GETRSP() (in file IOFUNCT.FOR)
Transforms a yes/Yes or no/NO response to a logical true and false.

CLOOK() (in file IOFUNCT.FOR)
Searches location of a 4-digit code in a set of string array.

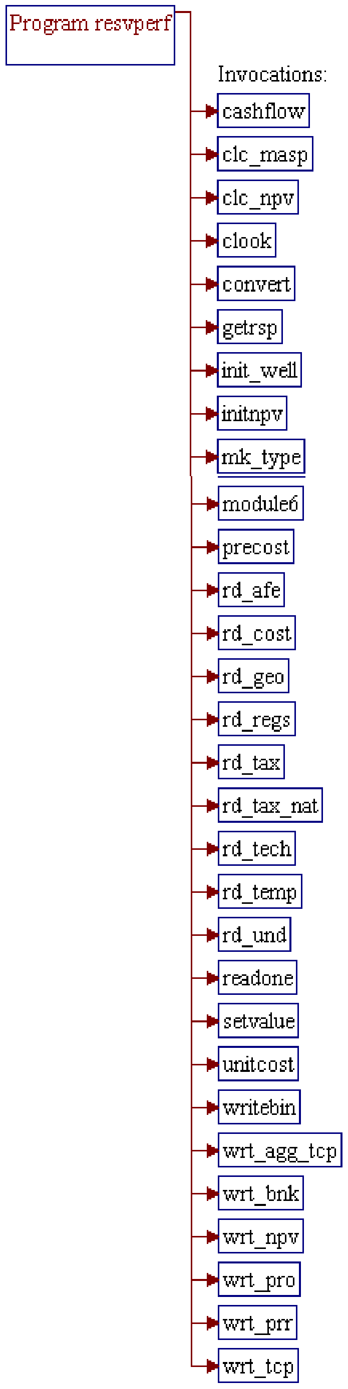
CALLED BY: None

READS: RUNSET.DAT
(Data of run specifications for the RP Module)
PLY_DFN.DAT
(Data of play level federal lands percentage for undiscovered reservoirs, impurity levels, etc.)
[GSAM].BIN

(Flow rate, pressure, time step, etc. data from previous type curve runs)

CREATES: [GSAM].CUR
(Summary of current technology)
[GSAM].SUM
(Summary of current technology with NPV's of drilling costs and total tax paid)
[GSAM].ADV
(Summary of advanced technology)
[GSAM].ENV
(Reservoir specific data to be used for environmental costing assignments)

ROUTINE INTERACTIONS:



Step 1: **Name of the main program for the RP Module is declared. Header ".h" files are included and local variables and common blocks are defined.**

Note: The name of the main program of the RP Module is RESVPERF. The following statement declares the program name.

```
      PROGRAM RESVPERF
```

Note: The include ".h" files consist of shared variable declarations and common blocks. Some of these header files are also included in the sub-programs of the RP Module for the purpose of sharing data between the caller and calling routines.

```
      include 'dimen.h'
      include 'global.h'
      include 'cashflow.h'
      include 'costing.h'
      include 'cost.h'
      include 'field.h'
      include 'tax_nat.h'
      include 'tax_reg.h'
      include 'unitcost.h'
      include 'gsamvar.h'
      include 'welldata.h'
      include 'type_out.h'
      include 'tech.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
      include 'npv.h'
```

Note: Local variables and common blocks are defined.

```
      integer ireg,irec,iyrenv,ich,nyrsi,imstart,rsty1
      logical matched,ihist,iundisc,ienvrun,envund,getrsp
      character*3 resp
      character*20 technmst,nname
      real*4 tgasb(qcase,qpay),contfact,roy
      common /prod_life/ iattt(qcase,qpay)
      common /history/ iprint
      common /stchg/ iwin_yr
      common /stchg1/ tgasb
```

Step 2: **Run specifications for the RP Module are read from input file RUNSET.DAT or from the computer keyboard (manual input).**

Note: The input file RUNSET.DAT is opened. The first entry of this text file is read and stored in variable *irunset*.

```
      open(unit=87,file='runset.dat',status='old')
      read(87,*)irunset
```

Note: If the first entry of the RUNSET.DAT is an integer 1 then the remaining data is read from the file. Otherwise, the program will prompt the user to enter the necessary information using computer keyboard. In the case of *irunset=1*, the program will read the data file in a sequence of comment/header line(s) and data line. The first data, an answer to a question whether the .GSM files are one line format or not, is read. Response to this question is a toggle Y or N and is stored in variable *resp*. The character Y or N in *resp* is transformed to a logical *.true.* for Y or *.false.* for N using the GETRSP() routine and stored in variable *iundisc*.

```
      if(irunset.ne.0)then
        read(87,*)
        read(87,'(a)') resp
        iundisc = getrsp(resp)
```

Note: The starting year of the model is read and stored in variable *imstart*.

```
        read(87,*)
        read(87,*) imstart
```

Note: A response to the question whether the history check needs to be performed or not is read and is transformed to a logical true or false (stored in *ihist*) using the routine GETRSP().

```
        read(87,*)
        read(87,'(a)') resp
        ihist = getrsp(resp)
```

Note: Variable *envund* is defaulted to a logical *.false.* (initialization for a non-environmental RP run) and the difference between the year for environmental RP run and the starting year of the model is defaulted to 41 years (*iyrenv=41*). Note that the maximum number of years for the RP runs is 40 years. Therefore, the *iyrenv=41* is the initialization for a non-environmental RP run. Values for both of these parameters will be changed if the model is found to be an environmental RP run.

```
         envund=.false.
         iyrenv=41
```

Note: A "model correction year" (*corr_yr*) is read. The production history of the reservoirs will be offset by this value (0 or 1) depending upon the release year of NRG data. The value is *0* for 1993 release or undiscovered data, *1* for 1994 release (such as the case for Canadian data).

```
         read(87,*)
         read(87,*) corr_yr
```

Note: Index for the environmental RP run (Y or N) is read and is transformed to a logical true or false (stored in *ienvrun*) using the GETRSP() routine. If the run is intended to be an environmental RP run (index is Y or *ienvrun=1*), the remaining data in the RUNSET.DAT are read and the difference between the year for the environmental RP run and the starting year of the model (*iyrenv*) is assigned or calculated. Run specifications for the environmental RP run are the Y/N toggle (index of the environmental run for producing reservoir and is transformed to a logical variable *envund*) and the year for the environmental RP run (*iyrstenv*). Environmental data is specified in COST.DAT file. For undiscovered reservoirs, environmental costs are added from producing reservoir from the year specified in file RUNSET.DAT

```
         read(87,*)
         read(87,'(a)') resp
         ienvrun = getrsp(resp)
         if(ienvrun)then
           read(87,*)
           read(87,'(a)') resp
           envund = getrsp(resp)
           if(.not.envund) then
             iyrenv=1
           else
             read(87,*)
             read(87,*) iyrstenv
             iyrenv = iyrstenv – imstart
           endif
         endif
```

Note: If the first entry of the RUNSET.DAT is not an integer 1, the remaining data in the file will be ignored. The program will prompt the user to enter the necessary information using the computer keyboard and will close the file. The parameters for the run specifications entered using the computer keyboard are similar to those specified in the RUNSET.DAT and descriptions are specified in earlier steps.

```
      else
        write(6,*) '============================================'
        write(6,*)
        write(6,*) '       IS THIS SIMPLIFIED GSM FILE ??'
        write(6,*) '       IF SIMPLIFIED DATABASE ENTER YES'
        write(6,*) '       IF FULL DATABASE FILE ENTER NO'
        write(6,*) '    (ENTER YES FOR UNDISCOVERED APPL,CANADA) '
        write(6,*) ' '
        write(6,*) '============================================'
        write(6,*) ' '
        read(5,'(a)') resp
        iundisc = getrsp(resp)
        if (iundisc) then
          ihist =.false.
          goto 346
        endif
        write(6,*) 'Enter Model Start Year (1993 etc..)'
        read(5,*) imstart
        write(6,*) '============================================'
        write(6,*) ' '
        write(6,*) '     DO YOU WANT TO DO HISTORY CHECK ?  '
        write(6,*)    ' '
        write(6,*) ' ENTER YES IF YOU WANT TO HAVE HISTORY CHECK '
        write(6,*) '  ENTER NO IF YOU DO NOT WANT HISTORY CHECK '
        write(6,*)   ' '
        write(6,*) '============================================'
        write(6,*)  ' '
        read(5,'(a)') resp
        ihist = getrsp(resp)
 346    envund=.false.
        iyrenv=41
        write(6,*) 'Enter Corr.year variable  For Model Stabilization'
        read(5,*) corr_yr
        write(6,*) 'Is this an Environmental Run (YES/NO)'
        write(6,*)  ' '
        read(5,'(a)') resp
        ienvrun = getrsp(resp)
        iyrenv=0
        if(ienvrun)then
          write(6,*) 'Is this for Producing Reservoirs? (Y/N)'
          read(5,'(a)') resp
          envund = getrsp(resp)
          if(.not.envund) then
            iyrenv=1
          else
            write(6,*) 'Enter Calender Year for Environmental Regs'
            write(6,*)  ' '
            read(5,*) iyrstenv
            iyrenv = iyrstenv – imstart
          endif
        endif
      endif
      close(87)
```

Step 3: **Names of development types are assigned to string variable array *casename().***

Note: The string variable array *casename()* (defined in header file GLOBAL.H) is used in some sub-programs of the RP Module for reporting purposes.

```
      casename(1)='Primary'
```

```
        casename(2)='Refrac '
        casename(3)='Infill '
```

Step 4: **The input file REGIONS.DAT is opened, the data is read using sub-program RD_REGS(), and the input file is closed.**

Note: The REGIONS.DAT file contains information about the list of .GSM files to be run through the RP Module. The file also contains YES/NO switches as indicators for opening specific files for consistency checks. The reading data process is performed in the RD_REGS() sub-program which is located in program file GSAM_A.FOR. The file also contains a toggle aligned with each [GSAM].GSM file entry indicating whether type curve modules have to be run or not. If this switch is set to “NO”, the corresponding [GSAM].BIN file should exist for reading pressures and flow rates.

```
        open(unit=87,file='regions.dat',status='old')
        call rd_regs(87)
        close(87)
```

Step 5: **The input file TAX_NAT.DAT is opened, the data is read using sub-program RD_TAX_NAT(), and the input file is closed.**

Note: The TAX_NAT.DAT file located under directory [working directory]\DATA contains information about the national tax treatment assumptions. The sub-program RD_TAX_NAT() located in file READINP.FOR reads the national tax data, converts the rate units from percent to fraction, and transfers the taxes information to the RP Module through shared variables in header files TAX_NAT.H and TAX_REG.H.

```
        open(unit=87,file='data\tax_nat.dat',status='old')
        call rd_tax_nat(87)
        close(87)
```

Step 6: **The input file COST.DAT is opened, the data is read using sub-program RD_COST(), and the input file is closed.**

Note: The COST.DAT file located under directory [working directory]\DATA contains cost related information. The sub-program RD_COST() located in file READINP.FOR reads the cost data, converts the rate units from percent to fraction, and transfers the information to the RP Module through shared variables in header files COST.H, and FIELD.H.

```
      open(unit=87,file='data\cost.dat',status='old')
      call rd_cost(87)
      close(87)
```

Step 7: **The input file TEMPLATE.DAT is opened, the data is read using sub-program RD_TEMP(), and the input file is closed.**

Note: The file TEMPLATE.DAT, located under directory [working directory]\DATA, contains information on fluid and reservoir properties data, well data, field development, drive mechanism, module type and other type curve related data. This file serves as a template in creating an input file to be used in the type curve module. The sub-program RD_TEMP() located in file GSAM_A.FOR reads the data from the file, stores the data as texts into an array variable *lines()*, and passes these texts (*lines()*) to sub-program MK_TYPE() (the sub-program to generate the input file for type curve module).

```
      open(unit=87,file='data\template.dat',status='old')
      call rd_temp(87)
      close(87)
```

Step 8: **The input file AFE.DAT is opened, the data is read using sub-program RD_AFE(), and the input file is closed.**

Note: The AFE.DAT file, located under directory [working directory]\DATA, contains information on authorization for expenditure charges for a producer. These data are taken from various sources including Joint Association Survey publications, the 1997 Well Cost Study by Petroleum Services Association of Canada (PSAC), and other ICF statistical estimates. This file is not currently used in running the RP Module. It does, however, provide the user with the structure of costing for a completed producing well. The sub-program RD_AFE() located in file READINP.FOR reads the data from the file, converts the units from percent to fraction, and passes the data to the RP Module through shared variables in header file UNITCOST.H.

```
      open(unit=87,file='data\afe.dat',status='old')
      call rd_afe(87)
      close(87)
```

Step 9: **The input file GEOLOGY.DAT is opened, the data is read using sub-program RD_GEO(), and the input file is closed.**

Note: The GEOLOGY.DAT file, located under directory [working directory]\DATA, contains information on reservoir property distributions by pay grade. The sub-program RD_GEO() located in file READINP.FOR reads the data from the file and shares them through variables in header file GEOLOGY.H. Data transfer from the RD_GEO() routine to the RP Module is done in sub-program CONVERT() which utilizes the shared variables in the header file GEOLOGY.H.

```
      open(unit=87,file='data\geology.dat',status='old')
      call rd_geo(87)
      close(87)
```

Step 10: **The input file TAXES.DAT is opened, the data is read using sub-program RD_TAX(), and the input file is closed.**

Note: The TAXES.DAT file, located under directory [working directory]\DATA, contains information on state income taxes, oil and gas severance taxes, and ad-voleram taxes. The taxes data are obtained from state publications (Chamber of Commerce data) for the U.S. and from NEB (National Energy Board) publications for Canadian provinces. The sub-program RD_TAX() located in file READINP.FOR reads the data from the file, converts the units from percent to fraction, and transfers the data to the RP Module via shared variables in header file TAX_REG.H.

```
      open(unit=87,file='data\taxes.dat',status='old')
      call rd_tax(87)
      close(87)
```

Step 11: **The input file TECH.DAT is opened, the data is read using sub-program RD_TECH(), and the input file is closed.**

Note: The TECH.DAT file, located under directory [working directory]\DATA, contains information on number of technologies (1 or 2) and data specifications for each technology. The sub-program RD_TECH() located in file READINP.FOR reads the data from the file and transfers the data to the RP Module via shared variables in header files COST.H, GSAMVAR.H and TECH.H.

```
      open(unit=87,file='data\tech.dat',status='old')
      call rd_tech(87)
      close(87)
```

Step 12: **The input file PLY_DFN.SPC is opened, the data is read, and the input file is closed.**

Note: The PLY_DFN.SPC file located under directory [working directory]\..\EXPLPROD is opened and two header lines are read. This file contains federal percentages of an undiscovered play, impurity levels, etc.

```
      open(unit=87,file='..\explprod\ply_dfn.spc',status='old')
      read(87,*)
      read(87,*)
```

Note: The play specifications are read, units conversions are performed, and the data are stored in the following variables:

• *cplay()*	Play identifier (4-digit USGS play code)
• *nname*	Region name
• *istate*	State code (currently not used)
• *vscl1*	First technology (current) development success rate
• *vscl2*	Second technology success rate
• *vscl3*	Third technology (advanced) development success rate
• *rstyl*	Index for dominant resource type of the play
• *dpth*	Average depth of play
• *roys()*	Royalty rate of play (%)
• *frac_play()*	% of undiscovered play on Federal land
• *h2scon()*	mole fraction of H2S
• *co2con()*	mole fraction of CO2
• *n2con()*	mole fraction of N2

The RP Module can accept a maximum of 1500 play-specific data sets. The number of maximum data set is specified in DIMEN.H with variable *qplay*.

```
      do 535 ipa=1,qplay
        read(87,881,end=54) cplay(ipa),nname,istate,vscl1,vscl2,
     @  vscl3,rsty1,dpth,roy,frac,h2scon(ipa),co2con(ipa),n2con(ipa)
881     format(a4,t21,a20,i2,3f6.1,i2,f9.1,f6.1,f8.2,f8.0,f8.0,f8.0)
        roys(ipa) = roy/100.
        frac_play(ipa) = frac/100.
        h2scon(ipa)=h2scon(ipa)/100.
```

```
         co2con(ipa)=co2con(ipa)/100.
         n2con(ipa)=n2con(ipa)/100.
535   continue
```

Note: After the end of file is detected, the number of data set (*ntotplay*) in the PLY_DFN.SPC is determined by subtracting the data counter (*ipa*) by 1. The total number of data is then stored in variable *ntotplay* and the input file is closed.

```
54    ntotplay = ipa - 1
      close(87)
```

Step 13: **Number of years to be analyzed (*nyrset*) is verified.**

Note: Number of years in time horizon (*qyr*) is specified to be 140 in DIMEN.H. The *nyrset*, given in the input file REGIONS.DAT, has to be less than *qyr*.

```
      if(nyrset.ge.qyr)nyrset=qyr-1
```

Step 14: **Variable for total number of records in .GSM files (*irec*) and loop for region (*ireg*) are initialized.**

Note: Total number of records in the .GSM files (*irec*) is calculated inside the region loop (*ireg*). Number of regions (*nreg*) is determined in sub-program RD_REGS() by counting the number of .GSM files specified in the input file REGIONS.DAT. The *ireg* loop is repeated *nreg* times.

```
      irec=1
      do 223 ireg=1,nreg
```

Step 15: **Index for printing header lines of output files .CUR and .ADV (*itt*) is initialized.**

Note: Header lines for output files .CUR (file unit #36) and .ADV (file unit #37) are printed only once. The index *itt* is used to indicate whether the header lines have been printed (*itt=1*) or not (*itt=0*).

```
       itt = 0
```

Step 16: **The .GSM file for the region designated by the loop counter *ireg* is opened and several files are opened for input/output.**

Note: Index for opening input/output files (*iz*) is initialized.

```
         iz = 0
```

Note: Name of the .GSM file of the loop counter *ireg* is extracted from the string variable *regnm()* (names of the .GSM files are assigned to the *regnm()* in sub-program RD_REGS()). This is done by locating the first white space in the *regnm()* (*ijk*) and taking the first *ijk-1* characters of the *regnm()*. These characters are then stored in a string variable *filenm*. Note that the length of the .GSM file name is assumed to be less than 8 characters.

```
         do 198 ijk = 1,8
           if (regnm(ireg)(ijk:ijk).eq.' '.and.iz.ne.1) then
             filenm=regnm(ireg)(1:ijk-1)
```

Note: The .GSM file of the current loop counter *ireg* (i.e. the counter for number of .GSM files to be processed) is opened.

```
             open(unit=10,file=regnm(ireg)(1:ijk-1)//'.gsm',status='old')
```

Note: The variable *runtype()* (see sub-program RD_REGS() and input file REGIONS.DAT) is used to indicate whether the current region (region *ireg*) requires the type curve runs. This logical variable stores the YES/NO indicators of the "Region" section in the REGIONS.DAT. A true condition (*runtype()* equals to *.true.* for YES) will open an output file .BIN to store the type curve results. The false condition (*runtype()* equals to *.false.* for NO) will open an already existing corresponding the input file .BIN to obtain the pre-run type curve data. The name of the .BIN file is the name of the corresponding .GSM file.

```
             if(runtype(ireg)) then
               open(unit=11,file=regnm(ireg)(1:ijk-1)//'.bin',
     @         form='unformatted')
             else
               open(unit=11,file=regnm(ireg)(1:ijk-1)//'.bin',
     @         status='old',form='unformatted')
             endif
```

Note: Several output files are opened. The names of the output files are the name of the corresponding .GSM file. These files are:

• .ERR	Error messages related to reading the .GSM files
• .PRD	Production and operating costs of each reservoir
• .CUR	Summary of current technology
• .ADV	Summary of advanced technology without detail on pay grade
• .DEC	Reservoir decisions and summary of economics
• .SUM	Summary of current technology which also reports the NPV's of drilling costs and total taxes
• .ASM	Summary of advanced technology for all the pay grades of the GSAMID
• .ENV	Environmental related data such as impurity concentrations and condensate yield.

```
            open(unit=9,file=regnm(ireg)(1:ijk-1)//'.err')
            open(unit=34,file=regnm(ireg)(1:ijk-1)//'.prd')
            open(unit=36,file=regnm(ireg)(1:ijk-1)//'.cur')
            open(unit=37,file=regnm(ireg)(1:ijk-1)//'.adv')
            open(unit=69,file=regnm(ireg)(1:ijk-1)//'.dec')
            open(unit=70,file=regnm(ireg)(1:ijk-1)//'.sum',recl=240)
            open(unit=79,file=regnm(ireg)(1:ijk-1)//'.asm',recl=240)
            open(unit=23,file=regnm(ireg)(1:ijk-1)//'.env',recl=240)
```

Note: Index for opening input/output files (*iz*) is set to 1 to indicate that the files are opened. The loop for extracting the .GSM file name (locating the white space) is closed.

```
            iz = 1
          endif
198     continue
```

Step 17: **Header lines for output files .SUM and .ASM are written.**

```
        write(70,*) 'Primary Wells Only (Current Tech.)'
        write(70,123)
123     format(1x,'GSAMID',8x,'Pay',4x,'Res.',6x,'OGIP',
     @  4x,'# Wells',3x,'MASP',6x,'NPV DRL',
     @  4x,'NPV Tax',7x,'Tax Diff')
        write(79,*) 'Primary Wells Only (Adv. Tech.)'
        write(79,123)
```

Step 18: **Line number for the top loop of the .GSM record (line number 444) is assigned.**

Note: The .GSM record loop (*irec*) is a nested loop inside the region loop (*ireg*). This loop is repeated using a "*goto 444*" statement located several lines before the end of the region loop (located at the bottom part of the RESVPERF.FOR program) until the "end of file" pointer of the .GSM file is encountered. Note that the statement "*continue*" in line number *444* is not paired to any "*do*" statement and it has no effect on the program. It just serves as a pointer for the *irec* loop indicating how many reservoirs have been processed so far (for the .GSM files located in REGIONS.DAT file)

```
444     continue
```

Step 19: **Index for discounted cash flow analysis (*maxcf*) is initialized.**

Note: The *maxcf* is initially set to zero as indicator for non-environmental RP run. This variable will be set to one if the run is found to be an environmental RP run.

```
        maxcf = 0
```

Step 20: **Database type of the .GSM file is checked (one-line format for undiscovered .GSM files and Appalachia) or full database format, i.e. NRG specified data), and one data record (i.e. one full entry for a GSAMID) is read from the .GSM file. The program flow is switched to line *222* if the "end of file" pointer of the .GSM file is encountered.**

Note: If the database type is a one-line format (*iundisc=.true.*), the sub-program RD_UND() is used to read the .GSM file; otherwise, the sub-program READONE() will be used. These sub-programs, located in program file READONE.FOR, will indicate the main program to switch the program flow to line *222* if the "end of file" pointer of the .GSM file is encountered.

```
        if (iundisc) then
          call rd_und(regnm(ireg),10,*222)
          goto 345
        endif
        call readone(9,10,ihist,*222)
```

```
345     continue
```

Step 21: **Index for history match (*matched*) is initialized.**

Note: The index *matched* is initially set to *.false.* to indicate that the field is not yet history matched.

```
         matched = .false.
```

Step 22: **Index of the play code (*ipa*) of the play array (from PLY_DFN.SPC input file) is determined.**

Note: The sub-program CLOOK() (in file IOFUNCT.FOR) returns the index of the play code (*ipa*) by comparing the play code in GSAMID (i.e. digits 5 through 8) with the codes stored in array *cplay()*. The *ntotplay* is the total number of plays in the PLY_DFN.SPC. The sub-program CLOOK() will return an *ipa* of zero if no match is found.

```
         call clook(gsamid(5:8),cplay,ntotplay,ipa)
```

Step 23: **An error message is printed to the screen and the RP run is halted if the play code as specified the current GSAMID is not in the play list (PLY_DFN.SPC input file).**

```
         if (ipa.eq.0) then
           print *, 'No Match for play in PLY_DFN.SPC file'
           print *, 'Check \explprod\ply_dfn.spc file'
           stop
         endif
```

Step 24: **Variables for the royalty rate (*royrate*) and percentage of play on Federal land (*frac_fed*) for the current GSAMID are set to the corresponding information obtained from the play list in Step 22.**

Note: The variable for percentage of play on Federal land (*frac_fed*) is assigned the value if the status of the current GSM ID is undiscovered (the third digit of the GSAMID is equal to 1).

```
          royrate = roys(ipa)
          if (gsamid(3:3).eq.'1') frac_fed = frac_play(ipa)
```

Step 25: **The impurity levels for H2S, CO2, and N2 (*h2s, co2, n2*) are set to those from the PLY_DFN.SPC input file only if the impurity levels information are not given in the .GSM file (summation of the impurity levels are less or equal to zero). In the case when impurity levels are not available in [GSAM].GSM files, play average impurity levels are assigned to the GSAMID as shown below.**

```
          if ((co2+n2+h2s).le.0.0) then
            h2s = h2scon(ipa)
            co2 = co2con(ipa)
            n2  = n2con(ipa)
          endif
```

Step 26: **The record number (*irec*), GSAMID (*gsamid*) and region's name (*regnm(ireg)*) for the currectly active-record (i.e. reservoir) are printed to the computer monitor.**

Note: The purpose of printing these data to the computer monitor is to inform the user about which GSAMID is currently processed by the RP Module and also to indicate that the RP Module is still running.

```
          write(6,'(t2,a,i4,a,a,a,a)')
     &    'Analyzing: ',irec,'  Gsam Code: ',gsamid,
     &    ' Region: ',regnm(ireg)
```

Step 27: **Variable *irec* is incremented which keeps track number of reservoirs processed.**

```
          irec=irec+1
```

Step 28: **Report files, as requested in input file REGIONS.DAT, are opened.**

Note: Each "YES" or "NO" flag under the question "Reports to Print" in the input file REGIONS.DAT, is translated into a logical .true. or .false. in the sub-program RD_REGS(). If the answer to generate a specific report is "YES", one file for each reservoir will be opened for that particular report. The file name will be the

GSAMID of the reservoir. The extension of the files and the corresponding logical variables are as follows:

• .TCO, *l_tco*	Type curve output file
• .TCI, *l_tci*	Type curve input file
• .PRO, *l_pro*	Detailed cash flow pro-forma file
• .NPV, *l_npv*	Net present value summary file
• .PRR, *l_prr*	Reduced form cash flow pro-forma file

```
      if(l_tco)then
        open(unit=55,file=gsamid(4:11)//'.tco')
      endif
      if(l_tci)then
        open(unit=56,file=gsamid(4:11)//'.tci')
      endif
      if(l_pro) then
        open(unit=31,file=gsamid(4:11)//'.pro')
      endif
      if(l_npv)then
        open(unit=33,file=gsamid(4:11)//'.npv')
      endif
```

Note: In this step, two header lines are printed to the output file .PRR (if the report is requested) immidiately after opening the file.

```
      if(l_prr) then
        open(unit=67,file=gsamid(4:11)//'.prr')
        write(67,'(20a)')
   &    '                                                   ',
   &    '                                               @ $2/Mcf ',
   &    '                                @ $5/Mcf '
        write(67,'(20a)')
   &    '                                                   ',
   &    '                 #            Tot.   Undisc.  Disc. ',
   &    ' Undisc.  Disc.   Tot.   Undisc.  Disc.  Undisc.  Disc.'
        write(67,'(20a)')
   &    ' GSAMID     Tech.          P.G.  Dev. Case.         Resv.',
   &    '   OGIP  Wells  MASP   Cap.   AT Cash AT Cash BT',
   &    ' Cash BT Cash  Cap.   AT Cash AT Cash BT Cash BT Cash'
      endif
```

Step 29: **Loop for the technology (*itech*) is initialized.**

Note: Counter for *itech* loop is set to the number of technologies (*ntech*) obtained from input file TECH.DAT.

```
      do itech=1,ntech
```

Step 30: **Variables for type curve calculation are initialized and data in the .GSM file are converted to type curve variables.**

Note: Window year (*iwin_yr*) is initialized to –1 as an indicator that the value is not yet calculated. *icounter* is used as a counter for reporting results consistently. *pg1fact* and *pg3fact* are factors that are used by sub-program CONVERT() to reduce or increase drainage area of pay grades 1 and 3. Currently these factors are not implemented (set to be zero).

```
      iwin_yr=-1
      icounter = 0
      pg1fact=0.0
      pg3fact=0.0
```

Note: Sub-program INIT_WELL is called. This routine initializes type curve and economics variables.

```
      call init_well
```

Note: *nyr*, number of years to be analyzed, is first set to 40. For Appalachia region, the *icounter* is calculated and added to the *nyr*. For Appalachia, *icounter* is a representation for historical years of production.

```
      nyr = 40
      if (regnm(ireg)(1:3).eq.'APP'.or.
     @    regnm(ireg)(1:3).eq.'app') then
        icounter = abs(imstart - istartappl-nint(corr_yr))
        if(icounter.le.1)icounter=1
        nyr = 40 +icounter
      endif
```

Note: Sub-program CONVERT() is called. This routine converts the .GSM data into pay grade level variables that feed to the type curve routines.

```
      call convert(itech,pg1fact,pg3fact)
```

Step 31: **The history match procedure is performed if it is requested in the RUNSET.DAT.**

Note: If the switch for history match in the RUNSET.DAT is “NO” (*ihist=.false.*), the indicator for performing history matched (*matched*) is set to *.false.* to proceed with type curve calculations (i.e. statement number 109) without history match procedure.

```
      if (.not.ihist) then
        matched = .false.
        goto 109
      endif
```

Note: Sub-program SETVALUE() is called. This routine predicts the production years of the reservoir prior to the year 1993.

```
      call setvalue(matched,itime,ireg)
```

Note: If the production rate in the year 1993 is less than or equal to 0.002 BCF/year, the type curve calculation is proceeded without history match procedure. The *icounter* is set to zero for the case of production rate less than 0.002 BCF/year or it is set to one otherwise.

```
      if (gasprd93.le.0.002) then
        matched = .false.
        icounter = 0
        goto 109
      endif
      icounter = 1
```

Step 32: **Type curves output for the reservoir in the current .GSM record are generated by sub-program MODULE6() or retrieved from binary file .BIN.**

Note: In this step the type curve outputs are obtained either by invoking the sub-program MODULE6() or reading the previously generated .BIN file. The MODULE6() is called if the type curve run is requested in the input file REGIONS.DAT. Prior to invoking the MODULE6(), the sub-program MK_TYPE() is called only when the input file for the type curve module is requested in the input file REGIONS.DAT. The following are descriptions of the type curve data read from the [GSAM].BIN file:

• *gsamid*	11-digit GSAM identification number of the reservoir
• *tgasb(qcase,qpay)*	Total recoverable gas reserves (BCF)
• *type_ogip(qcase,qpay)*	Original gas in place (BCF)
• *type_well(qcase,qpay)*	Number of total wells based on spacing constraints
• *kwinyr(qcase,qpay)*	Window year (years)
• *iattt(qcase,qpay)*	Total productive years (years)
• *type_gas(qcase,qpay,qyr)*	Gas production (BCF/year)

- *type_pwhp(qcase,qpay,qyr)* Primary wellhead pressure (psia)

```
109      if(runtype(ireg)) then
           if(l_tci) call mk_type(56,itech)
           call module6(filenm,nyr,1,3,l_tco,matched)
         else
           read(11,err=1)gsamid,tgasb,type_ogip,type_well,kwinyr,iattt,
     @     type_gas,type_pwhp
         endif
```

Step 33: **Window year (*iwin_yr*) is calculated.**

Note: The window year (*iwin_yr*) is the number of years for which the total flow rate of the reservoir (from the three pay grades) remains constant. In the type curve module (*MODULE6()*), both infill drilling and refracturing cases are implemented when the sandface pressure reaches the minimum allowable value (the value specified in the input file TECH.DAT). In the following section, *iwin_yr* is calculated by searching for the year (starting from year 2) when the difference between the production of the primary case (*type_gas(1,...,...)*) and the infill case (*type_gas(3,...,...)*) is greater than 0.002 BCF. Note that productions from these two development cases will be the same if the infill drilling is not yet implemented. The variable *ich* is the year in which gas production due to infill drilling is higher than primary drilling by at least 0.002 BCF/year. This indicates that infill drilling produces more than primary drilling in year *ich*. The variable *iwin_yr* is then calculated by subtracting *icounter* from *ich* and the value is forced between 2 and *qyr*. *icounter* variable makes sure that years are reported consistently for all [GSAM].GSM files.

```
         do ich=2,nyr
           checkgas=0.0
           do ipa=1,3
             checkgas=checkgas+(type_gas(3,ipa,ich)-
     @         type_gas(1,ipa,ich))
           end do
           if(checkgas.ge.0.002) go to 9937
         end do
9937     iwin_yr=ich-icounter
         if(iwin_yr.gt.nyr) iwin_yr=qyr
         if(iwin_yr.le.2) iwin_yr=2
```

Step 34: **Peak production rate (*peakrate*) from the reservoir with primary development case is calculated.**

Note: The peak rate calculation is skipped if the reservoir is a water drive gas reservoir (*module* equals to 5) with automatic refracturing. The peak rate is defined as the total production rate (from the three pay grades with primary development type) divided by the number of wells in the first year. It is assumed that production rate in the first year is the highest.

```
          if (module.eq.5.and.icase.eq.2) goto 987
          peakrate=
     &    (type_gas(1,1,1)+
     &    type_gas(1,2,1)+
     &    type_gas(1,3,1))/
     &    (type_well(1,1)+
     &    type_well(1,2)+
     &    type_well(1,3))/365*1.e6
```

Step 35: **Main loop for development types (*icase=1,2* and *3*) and pay grades (*ipay=1,2* and *3*) starts. These loops are within the technology loop of Step 29.**

Note: All economic calculations are performed within these two nested loops. The inner loop is the pay grade loop (*ipay*) and the outer loop is the development type loop (*icase*). The following codes initialized the two loops.

```
987       do icase = 1,3
            do ipay=1,3
```

Note: Prior to the NPV calculations, all variables for NPV and slope of NPV are initialized to zeros. The variables are:

• *npv_prd()*	NPV of gas production
• *npv_exp()*	NPV of total expenses
• *npv_inv()*	NPV of total investments
• *npv_drl()*	NPV of drilling costs
• *npv_tax()*	NPV of taxes
• *slope1()*	Slope of NPV due to change only in drilling cost (drilling slope)
• *slope2()*	Slope of NPV due to changes in all non drilling costs (non-drilling slope)

```
              npv_exp(1,icase,ipay)=0.0
              npv_exp(2,icase,ipay)=0.0
              npv_inv(1,icase,ipay)=0.0
              npv_inv(2,icase,ipay)=0.0
              npv_drl(1,icase,ipay)=0.0
              npv_drl(2,icase,ipay)=0.0
              npv_tax(1,icase,ipay)=0.0
              npv_tax(2,icase,ipay)=0.0
              slope1(icase,ipay)=0.0
              slope2(icase,ipay)=0.0
```

Note: Cash flow variables are initialized. For environmental RP run, *maxcf* variable is set to one otherwise it is zero.

```
                if(ienvrun)maxcf=1
                tot_cap_2(icase,ipay)=0.0
                udatcf_2(icase,ipay)=0.0
                datcf_2(icase,ipay)=0.0
                udbtcf_2(icase,ipay)=0.0
                dbtcf_2(icase,ipay)=0.0
```

Note: Sub-program INITNPV is invoked to initialize NPV related variables.

```
                call initnpv
```

Note: Number of wells ini each pay grade is assigned to variable *nwell*. It is assumed that number of wells in the primary development type would be used for this assignment.

```
                nwell=type_well(1,ipay)
```

Note: Number of years (nyr) is initialized to 40 years irrespective of value specified in input file RUNSET.DAT.

```
                nyr = 40
```

Step 36: **In this step, methane gas production (*gasprod()*) in each year is calculated, gas price (*gprice()*) and total operating and maintenance (O&M) cost (*totoam()*) in each year are initialized, and number of production years (*nyr*) for which gas production rate is greater than 0.001 BCF/year is determined and stored in *nyr* variable.**

Note: The methane gas production (*gasprod()*) is calculated by subtracting the impurities (CO2, N2, and H2S) from the net gas production (*type_gas()*). The gas price (*gprice()*) is set to $2/MCF and the total O&M cost is initialized to zero. The above calculations/assignments are performed inside the year loop *iyr*. The *iyr* loop is terminated when the gas production is less than 0.001 BCF (the reservoir is depleted). At the end of the *iyr* loop, the number of years the reservoir is produced (*nyr,nyrsi*) is calculated.

```
               do 111 iyr=1,nyr
                 gasprod(iyr)=type_gas(icase,ipay,iyr+icounter)
     @           *(1.0-co2-n2-h2s)
                 gprice(iyr)=2.0
                 totoam(iyr)=0.0
                 if (gasprod(iyr).lt.0.001) then
                   goto 113
                 endif
111              continue
113            nyr = iyr – 1
               if (nyr.lt.1) nyr = 1
               nyrsi=nyr
```

Step 37: **The first cash flow calculations (at $2/MCF gas price, the base case scenario) are performed.**

Note: Sub-program UNITCOST() is invoked to calculate unit costs.

```
               call unitcost(ktech)
```

Note: Sub-program PRECOST() is called. This routine uses the unit cost data to create the cost streams to be fed to the cash flow routine (CASHFLOW()).

```
               call precost(ktech,icase,iyrenv)
```

Note: Sub-program CASHFLOW() is invoked to perform a discounted pro-formacash flow analysis.

```
               call cashflow(ktech,nyrsi,maxcf)
```

Note: Sub-program CLC_NPV() is invoked to perform NPV calculations.

```
               call clc_npv(1,ktech)
```

Note: Total capital investment, discounted and undiscounted cash flows (before and after tax) and total O&M cost for the gas price of $2/MCF are calculated for all the years (*nyr*). The total O&M includes general O&M, environmental O&M, severance tax, and royalties.

```
               do iyr=1,nyr
                 tot_cap_2(icase,ipay)=tot_cap_2(icase,ipay)+
     &           ti(iyr)+ii(iyr)
                 udatcf_2(icase,ipay)=udatcf_2(icase,ipay)+
     &          aatcf(iyr)
                datcf_2(icase,ipay)=datcf_2(icase,ipay)+
     &          datcf(iyr)
                udbtcf_2(icase,ipay)=udbtcf_2(icase,ipay)+
     &          nibt(iyr)
                dbtcf_2(icase,ipay)=dbtcf_2(icase,ipay)+
     &          nibt(iyr)/((1+disc)**(iyr-1))
                totoam(iyr)=oam(iyr)+eoam(iyr)+sevtax(iyr)
     &          +adjgross(iyr)*royrate
              end do
```

Step 38: **Output files of the summary of the first cash flows (at $2/MCF gas price) are generated.**

Note: Sub-program WRT_PRO() is called. This routine writes out cash flow pro-forma to output file .PRO. The .PRO output file is generated only if the report is requested in the input file REGIONS.DAT. This reporting process is performed only for the base case which is for pay grade 2 (*ipay=2*) with current technology (*ktech=1*), and with primary development type (*icase=1*).

```
              if(l_pro.and.ipay.eq.2 .and.ktech.eq.1 .and. icase.eq.1)
     &        call wrt_pro(31,ktech,icase,ipay)
```

Note: Sub-program WRT_TCP() is invoked to write out production and operating costs to output file .PRD for both technology cases. Prior to this, the technology names (*technm()*) are assigned.

```
              if (envund.and.itech.ge.2) then
                technmst=technm(ktech)
                technm(ktech)=technm(itech-1)
                technm(ktech)=technmst
              endif
              if (itech.le.2) then
                call wrt_tcp(34,ktech,icase,ipay,icounter)
              endif
```

Step 39: **The second cash flow calculations (at $5/MCF gas price) are performed.**

Note: Gas and oil prices are readjusted to the $5/MCF gas price. The unit for oil is converted from MCF to BBL.

```
              do iyr=1,nyr
                gprice(iyr)=gprice(iyr)+3.0
                oprice(iyr)=oprice(iyr)+5.642*3.
              end do
```

Note: The sub-programs UNITCOST(), PRECOST(), CASHFLOW(), and CLC_NPV() are invoked to perform similar procedures as in the first cash flow calculations.

```
        call unitcost(ktech)
        call precost(ktech,icase,iyrenv)
        call cashflow(ktech,nyrensi,0)
        call clc_npv(2,ktech)
```

Step 40: **The third cash flow calculations are performed at $2/MCF gas price and zero drilling costs.**

Note: Gas and oil prices are set back the original values (i.e. $2/MCF gas price).

```
        do iyr=1,nyr
          gprice(iyr)=gprice(iyr)-3.0
          oprice(iyr)=oprice(iyr)-5.642*3.0
        end do
```

Note: The sub-program UNITCOST() is invoked for unit costs calculations.

```
        call unitcost(ktech)
```

Note: Development drilling cost (*dwc_w*), exploration drilling cost (*ewc_w*), stimulation cost (*stim_w*), and compressor cost (*comp_w*) are initialized to zeros.

```
        dwc_w= 0.0
        ewc_w = 0.0
        stim_w=0.0
        comp_w=0.0
```

Note: The sub-programs PRECOST(), CASHFLOW(), and CLC_NPV() are invoked to perform similar procedures as in the previous cash flow calculations.

```
        call precost(ktech,icase,iyrenv)
        call cashflow(ktech,2,0)
        call clc_npv(3,ktech)
```

Step 41: **The fourth cash flow calculations are performed at $2/MCF gas price with all other costs set to zero, i.e. by setting all non-drilling costs to zero.**

Note: The sub-program UNITCOST() is invoked for unit costs calculations.

```
          call unitcost(ktech)
```

Note: Setting all non-zero costs (per well basis) equal to zeros. These costs are:

• *fac_w*	Facilities cost
• *fxoam_w*	Fixed O&M cost
• *h2ooam*	Surface O&M water cost
• *voam_g*	Surface O&M gas cost
• *comp_oam*	Compressor O&M cost
• *env_oam_g*	Environmental O&M gas cost
• *env_oam_w*	Environmental O&M water cost
• *env_oam_l*	Environmental O&M well cost
• *env_oam_n*	Environmental O&M gas new well cost
• *lbc_frac*	Lease bonus cost factor

```
          fac_w=0.0
          fxoam_w=0.0
          h2ooam_w=0.0
          voam_g=0.0
          comp_oam=0.0
          env_oam_g=0.0
          env_oam_w=0.0
          env_oam_l=0.0
          env_oam_n=0.0
          lbc_frac=0.0
```

Note: The sub-programs PRECOST(), CASHFLOW(), and CLC_NPV() are invoked to perform similar procedures as in the previous cash flow calculations.

```
          call precost(ktech,icase,iyrenv)
          call cashflow(ktech,4,0)
          call clc_npv(4,ktech)
```

Step 42: Sub-program WRT_NPV() is called.

Note: In sub-program WRT_NPV(), the NPV's are written to output file .NPV. This process is performed only if the .NPV report is requested in the input file REGIONS.DAT.

```
               if(l_npv) call wrt_npv(33,ktech,icase,ipay)
```

Step 43: **The NPV's and slope of NPV's defined in Step 35 are assigned.**

Note: The assignments are performed only when the NPV of gas production for the base case (*g_prd_npv(1)*) is not zero (greater than 0.0001).

```
               if(g_prd_npv(1).ge.0.0001)then
                 npv_prd(icase,ipay)=g_prd_npv(1)
                 npv_exp(1,icase,ipay)=toc_npv(1)
                 npv_exp(2,icase,ipay)=toc_npv(2)
                 npv_inv(1,icase,ipay)=tot_inv(1)
                 npv_inv(2,icase,ipay)=tot_inv(2)
                 npv_drl(1,icase,ipay)=drl_inv(1)
                 npv_drl(2,icase,ipay)=drl_inv(2)
                 npv_tax(1,icase,ipay)=tax_npv(1)+credit_npv(1)
                 npv_tax(2,icase,ipay)=tax_npv(2)+credit_npv(2)
                 slope1(icase,ipay)=(npv(3)-npv(1))/(totalcst(4))
                 slope2(icase,ipay)=(npv(4)-npv(1))/(totalcst(3))
               endif
```

Step 44: **Sub-program CLC_MASP() is invoked.**

Note: The sub-program CLC_MASP() is invoked to calculate the Minimum Acceptable Supply Price (MASP) of the base price.

```
               call clc_masp(ktech,icase,ipay,iyrenv)
```

Step 45: **The loops declared in Step 35 are closed.**

```
              end do
           end do
```

Step 46: **Total methane gas produced in the first year i.e. year 1993 (*sumflow*), total methane gas produced throughout the 40 years of the production (*sumres*), and total number of wells (*nnwell*) for all the pay grades are calculated.**

Note: The methane gas production is calculated by subtracting the impurities (CO2, N2, and H2S) from the net gas production. The number of wells is calculated based on the primary producing wells (*type_well()* > 0).

```
         sumflow = 0.0
         sumres  = 0.0
         nnwell=0.0
         do 437 ipay =1, 3
            sumflow = sumflow + type_gas(1,ipay,1+icounter)
     @      *(1.0-co2-n2-h2s)
            if (type_gas(1,ipay,1+icounter).gt.0.0) then
              nnwell  = nnwell + type_well(1,ipay)
            endif
437      continue
         do 438 iyr = 1,40
           do 438 ipay = 1,3
              sumres = sumres + type_gas(1,ipay,iyr+icounter)
     @        *(1.0-co2-n2-h2s)
438      continue
```

Step 47: **Results from the current and advanced technologies are written to output files .CUR and .ADV.**

Note: Header lines for output files .CUR and .ADV are printed. The index *itt* which was previously initialized to zero indicates in Step 15 that the header lines are not yet printed. At the end of these instructions, the index *itt* is set to 1.

```
         if (itt.eq.0)  then
           write(36,*)
     @     'Summary of Primary Wells Only For Current Tech.'
           write(37,*)
     @     ' Summary of Primary Wells Only For Advanced Tech.'
           write(36,136)imstart,imstart
           write(37,136)imstart,imstart
         endif
136      format(1x,'GSAMID',7x,'STATE',3x,'DEPTH',4x,'MASP',5x,
     @   'Model',7x,'Reported',10x,'Teh.Rec.',4x,'Reported',
     @   5x,'Model',3x,'Reported',6x,'Model',8x,'Model',
     @   6x,'Total Proved','       CG9R93', '      RepOGIP',
     @   '    MOGIP'
     @   ,/,31x,'PG 2',
     @   '   ',i4,' Prod.',3x,i4,' Prod.',8x,
     @   'Prim. Res.',3x,'Tot.Res.',5x,'Wells',4x,'Wells',
     @   7x,'Spacing',5x,'Est. Area',9x,'Area')
         itt = 1
```

Note: Data of the current and advanced technologies are printed to [GSAM].CUR and [GSAM].ADV output files.

```
         if (ktech.eq.1) then
           write(36,447) gsamid,state,avdep,masp(1,2),sumflow,
     @     gasprd93,sumres,grsv93,nnwell,prdwel93,wlspac1,acprod,
     @     acprov,cgpr93,ogip,type_ogip(1,1)+type_ogip(1,2)+
     @     type_ogip(1,3)
         else
           write(37,447) gsamid,state,avdep,masp(1,2),sumflow,
     @     gasprd93,sumres,grsv93,nnwell,prdwel93,wlspac1,acprod,
     @     acprov,cgpr93,ogip,type_ogip(1,1)+type_ogip(1,2)+
     @     type_ogip(1,3)
         endif
447      format(a11,3x,i5,3x,f6.0,2x,f5.2,1x,f10.5,2x,f11.4,
     @   5x,f12.4,1x,f12.4,3x,i7,2x,i7,2x,f12.2,2x,f14.0,2x,
     @   f14.0,1x,f9.3,1x,f12.3,1x,f12.3)
```

Step 48: **Sub-program WRT_PRR() is called.**

Note: The sub-program WRT_PRR() is invoked only if a report of reduced form of cash flow pro-forma (.PRR file) is requested in the input file REGIONS.DAT.

```
        if(l_prr) call wrt_prr(67,ktech)
```

Step 49: **Sub-program WRT_BNK() is called.**

Note: The sub-program WRT_BNK() is invoked to report the reservres, OGIP, and summary economics in output file [GSAM].DEC for all pay grades (3), development types (3), and technologies (2). This sub-program also reports other summary data for current and advanced technologies in [GSAM].SUM and [GSAM].ASM files.

```
        if (itech.eq.1) call wrt_bnk(69,ktech,70)
        if (itech.eq.2) call wrt_bnk(69,ktech,79)
```

Step 50: **Sub-program WRITEBIN() is called.**

Note: If type curve run is requested in the input file REGIONS.DAT, (*runtype() = .true.*), the sub-program WRITEBIN() is invoked to write out type curve outputs to the output file .BIN.

```
        if (runtype(ireg)) call writebin(11,icounter)
```

Step 51: **The *itech* loop (technology loop), declared in Step 29, is closed**

```
      end do
```

Step 52: **Outputs to file .ENV are written.**

Note: Values for parameters in the environmental output file .ENV are printed. These parameters are:

• *gsamid*	11-digit GSAM identification number

• *state*	4-digit state code
• *avdep*	Depth (feet)
• *acprod*	Drainage area (acres)
• *royrate*	Royalty rate
• *frac_fed*	Percentage of reservoir on Federal lands. For undiscovered reservoirs it is play specific percentage (0-100%); for discovered producing reservoirs, it is either 0% or 100%.
• *co2*	CO2 content (fraction)
• *n2*	N2 content (fraction)
• *h2s*	H2S content (fraction)
• *nglfact*	Condensate yield (BBL/MCF)

```
        write(23,9773) gsamid,state,avdep,acprod,royrate,
     @  frac_fed,co2,n2,h2s,nglfact
9773    format(a11,i5,1x,f7.0,1x,f10.0,1x,f7.3,1x,f7.3,1x,f7.5,1x,
     @  f7.5,f7.5,1x,f11.3)
```

Step 53: **Output files .TCI (file unit #56), .PRO (file unit #31), .TCO (file unit #55), .NPV (file unit #33), and .PRR (file unit #67) are closed.**

Note: These output files are closed prior to processing the next record of the .GSM file.

```
        close(56)
        close(31)
        close(55)
        close(33)
        close(67)
```

Step 54: **The .GSM loop (declared in Step 18) is repeated to process the next reservoir of the .GSM file.**

```
        goto 444
```

Step 55: **Output files .GSM (file unit #10), .PRD (file unit #34), .DEC (file unit #69), .SUM (file unit #70), and .ASM (file unit #79) are closed.**

Note: These output files are closed prior to opening and processing the next .GSM file. Program line 222 in the following code serves as a designation for the program control if the "end of file" pointer of the .GSM file is encountered as mentioned in Step 20.

```
222     close(10)
        close(34)
        close(69)
        close(70)
        close(79)
```

Step 56: **The region loop (*ireg*), declared in Step 14, is repeated to read and process the next .GSM file specified in the input file REGIONS.DAT.**

```
223   continue
```

Step 57: **Output files .PRD (file unit #34), .DEC (file unit #69), and .ENV (file unit #23) are closed.**

Note: These output files are closed prior to opening and processing the next .GSM file. Program line 222 in the following codes serves as a designation for the program control if the "end of file" pointer of the .GSM file is encountered as mentioned in Step 20.

```
      close(34)
      close(69)
      close(23)
```

Step 58: **The main program RESVPERF is complete.**

Note: The main program RESVPERF is completed and the program is terminated.

```
      stop
      end
```

SUB-PROGRAM RD_AFE()

LOCATION: READINP.FOR

MAIN THEME: This routine reads input file AFE.DAT which contains information on percentages of investment in normal AFE (authorization for expenditure) categories.

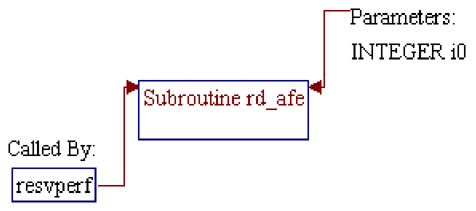
CALLS: None

CALLED BY: RESVPERF (in file RESVPERF.FOR)
Main program of Reservoir Performance Module.

READS: AFE.DAT
(Percentages of investment in normal AFE (authorization for expenditure) categories)

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameter of the sub-program are declared. Header ".h" files are included and local variable is defined.**

Note: Name of the sub-program is RD_AFE() and the parameter passed to this sub-program is as follows:

- *i0* unit number for input file AFE.DAT.

```
      subroutine rd_afe(i0)
```

Note: Header ".H" files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'unitcost.h'
```

Note: Local variable is defined.

```
      integer i0
```

Step 2: **Header lines are read from the file.**

```
      read(i0,*)
      read(i0,*)
      read(i0,*)
      nafe=1
```

Step 3: **Reads various AFE proportion categories (*afename*) and % of total cost for the AFE category (*afe*) are read in the following section. The entries are not used in the RP module, however, it does provide the user with the structure of costing for a completed producing well.**

```
 100  read(i0,'(t1,a,t35,f6.0)',end=200) afename(nafe),afe(nafe)
      afe(nafe)=afe(nafe)/100
      nafe=nafe+1
      goto 100
 200  continue
      nafe=nafe-1
      return
      end
```

SUB-PROGRAM RD_COST()

LOCATION: READINP.FOR

MAIN THEME: This routine reads cost data from the COST.DAT file for the set of technologies specified. COST.DAT is intended to be used for changing the costing parameters of the Reservoir Performance Module, impacting the economics of a reservoir, and the subsequent decisions in the E&P Module. Many of the parameters may be altered for sensitivity analysis. In sensitivity analysis, although the cases must be named current and advanced, this does not necessarily mean that, for instance, advanced must model an advanced technology. A user could, if desired, change the horizontal drilling cost in one region, and have all the information the same in the COST.DAT file, to model sensitivity to the cost of drilling a horizontal well.

A note on the functioning of the RP and E&P Modules: The costs and financial information used in the RP Module (such as NPV of investment, NPV of expenses, NPV of drilling costs, NPV of non-drilling costs, etc.) are stored in the .DEC file. The E&P Module performs a linear interpolation/extrapolation of these numbers at a specified gas price. This is possible because the RP uses flat gas prices of $2/Mcf and $5/Mcf, using $2 for the NPV calculations. These computations are then updated at the specified gas price track in the E&P Module.

The COST.DAT file is normally set up for two technologies: current and advanced. Under advanced technology several assumptions have been made, and may be changed if desired. Facilities well costs are improved by 20%, drilling costs by 10%, and compressor O&M by 1%.

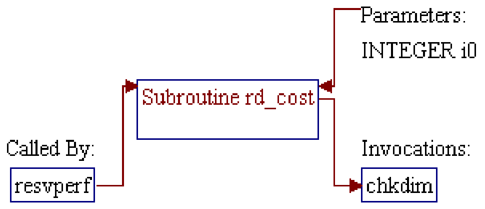
CALLS: CHKDIM() (in file IOFUNCT.FOR)
Checks if dimension of an array has been exceeded.

CALLED BY: RESVPERF (in file RESVPERF.FOR)
Main program of Reservoir Performance Module.

READS: COST.DAT
(Cost related information)

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameter of the sub-program are declared. Header ".h" files are included. Local variables are declared.**

Note: Name of the sub-program is RD_COST() and the parameter passed to this sub-program is as follows:

- *i0* Unit number for input file COST.DAT

```
      subroutine rd_cost(i0)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'cost.h'
      include 'field.h'
```

Note: Local variables are declared.

```
      integer i0,istep,itech,ncase
```

Step 2: **Discount rate (*disc*) reflects normal assumptions about inflation and the opportunity cost of capital, and may be changed as desired.**

```
      read(i0,*)
      read(i0,*) disc
      disc=disc/100.0
```

Step 3: **Number of technology cases is set in the "Number of Technology Cases" line. This number may be either 1 or 2. Be sure that each technology case is named in the next line and dimensions are not exceeded.**

```
      read(i0,*)
      read(i0,*) ncase
      call chkdim(ncase,qtech,'qtech')
```

Step 4: **The read statements are invoked within the following do loop. The cost data are read and stored for *ncase* number of technologies specified in COST.DAT file. Name of the technology is read.**

```
       do itech=1,ncase
        read(i0,*)
        read(i0,'(a)') casenm(itech)
```

Step 5: **The exploratory well cost factor describes how the inherent characteristics of exploratory drilling make it more expensive than development drilling. NOTE however that the RP Module does not currently model the drilling of exploration wells (it is done in the E&P Module), so that this factor is NOT currently used in the RP model. A similar parameter is specified in the E&P Module's DRL_CST.SPC file, and is used in the E&P Module.**

```
        read(i0,*)
        read(i0,*) ewc_fac(itech)
```

Step 6: **The lease bonus cost factor is assumed to be a fraction of the total revenue that could be generated from the reservoir, and the lease bonus cost is calculated by multiplying the factor with the total collected revenue.**

```
        read(i0,*)
        read(i0,*) lbc_fac(itech)
```

Step 7: **The G&G costs are a fraction of the exploratory well costs. However, because the RP Module does not currently model exploratory drilling this factor is not presently employed.**

```
        read(i0,*)
        read(i0,*) gg_fac(itech)
```

Step 8: **Based on various reports, like the PSAC Well Cost Study, the dry hole cost is on average 70% of the total cost for a completed development well producer.**

```
        read(i0,*)
        read(i0,*) pdry_dev(itech)
        pdry_dev(itech)=pdry_dev(itech)/100
```

Step 9: **The tangible and intangible percentages are industry averages. It is assumed that surface facilities are 100% tangible for tax accounting purposes.**

```
        read(i0,*)
        read(i0,*) ewc_tan(itech)
        ewc_tan(itech)=ewc_tan(itech)/100
        read(i0,*)
        read(i0,*) dwc_tan(itech)
        dwc_tan(itech)=dwc_tan(itech)/100
        read(i0,*)
        read(i0,*) fac_tan(itech)
        fac_tan(itech)=fac_tan(itech)/100
```

Step 10: **The environmental capital cost multiplier is a percentage of the facilities cost. It is the base environmental cost. A 10% factor, for example, designates surface facilities installed that handle gas stream (including impurities), water production, etc. at 10% of the total surface facilities cost. The incremental environmental compliance costs in the E&P Module or the RP Module are on top of this base cost. The E&P compliance costs are incremental and are applied in the E&P Module through various files. The RP compliance costs (described below) are incremental but are constant through time.**

```
        read(i0,*)
        read(i0,*) eccm(itech)
```

Step 11: **The general and administration overhead expense and capital multipliers are set.**

```
        read(i0,*)
        read(i0,*) ga_exp_m(itech)
        read(i0,*)
        read(i0,*) ga_cap_m(itech)
```

Step 12: **The regional development well cost table contains entries that correspond to the coefficients of a polynomial regression equation that is the best fit of the historical cost vs. depth data from the 1997 JAS Survey. As such, these entries should not be altered unless a similar procedure is undertaken. The number of regions, excluding the default values (#99), must be set before these values. The drilling cost coefficient values appear in 4 columns and are in thousand dollars as a function of depth. The first cost column is the intercept and the next three are the coefficients of x^a where x is depth, in feet. The development drilling cost calculation is demonstrated in the following example:**

For region #1 the cost columns are:

27.068800   4.7098399e-2 -2.547277e-6 1.18087525e-10

So that:

$$cost = 27.07 + (4.71e^{-2})x + (-2.55e^{-6})x^2 + (1.18e^{-10})x^3$$

If depth x = 1000 feet then:

$$cost = 27.07 + (4.71e^{-2})(1000) + (-2.55e^{-6})(1000)^2 + (1.18e^{-10})(1000)^3 = 71.738$$

From these calculations, the cost of a development well in region #1 (Appalachia) at a depth of 1000 feet is $71,738. The column on the far right is a multiplier (of the development well drilling cost) for horizontal or vertical technology (1.3 means that horizontal wells are 130% as costly as vertical wells). The vertical well cost file should normally have 1's in this column. Note that this factor can be used for modeling any other technology such as costlier drilling mud (synthetic muds, sour formations, etc.)

Number of regions for which development drilling costs are specified is read and dimensions are checked. Entries are then read for all the regions and the default.

```
        read(i0,*)
        read(i0,*) ndwcreg(itech)
        call chkdim(ndwcreg(itech),qreg,'qreg')
        read(i0,*)
        do ireg=1,ndwcreg(itech)
         read(i0,*)
     &   dwc_reg(itech,ireg),dwck(itech,ireg),dwcx(itech,ireg),
     &   dwcxx(itech,ireg),dwcxxx(itech,ireg),dcstf(itech,ireg)
        enddo
         read(i0,*)
     &   dwc_reg(itech,qreg+1),dwck(itech,qreg+1),dwcx(itech,qreg+1),
     &   dwcxx(itech,qreg+1),dwcxxx(itech,qreg+1),dcstf(itech,qreg+1)
```

Step 13: **Environmental costs in general are specified in the E&P module. However, if the user wants to use environmental compliance costs from start of the run in the RP module, then they can be specified. The number of regions that have environmental costs must be specified. The entries are as follows :**

Column 1: GSAM supply region indicator
Column 2: Existing well environmental tangible capital cost (K$/Well), incremental
Column 3: Existing well environmental intangible capital cost (K$/Well), incremental
Column 4: Existing well environmental operating cost (K$/Well), incremental
Column 5: New well environmental tangible capital cost (K$/Well), incremental

Column 6: New well environmental intangible capital cost (K$/Well), incremental
Column 7: New well environmental operating cost (K$/Well), incremental
Column 8: Incremental environmental cost related to drilling, %/ft
Column 9: Incremental environmental cost related to gas production handling (impurities), $/MCF
Column 10: Incremental environmental cost related to associated water production, $/BBL

The year these environmental costs would be applicable is specified in RUNSET.DAT file.

```
      read(i0,*)
      read(i0,*)newcreg(itech)
      read(i0,*)
      if(newcreg(itech).le.40)then
      call chkdim(newcreg(itech),qreg,'qreg')
      ienvr=qreg
      else
      call chkdim(newcreg(itech),qstate-1,'qstate')
      ienvr=newcreg(itech)
      endif
      do ireg=1,newcreg(itech)
       read(i0,*)ewc_reg(itech,ireg),
    &   env_et(itech,ireg),env_ei(itech,ireg),env_ee(itech,ireg),
    &   env_nt(itech,ireg),env_ni(itech,ireg),env_ne(itech,ireg),
    &   env_nf(itech,ireg),env_g(itech,ireg),env_w(itech,ireg)
      enddo
      read(i0,*)ewc_reg(itech,ienvr+1),
    & env_et(itech,ienvr+1),env_ei(itech,ienvr+1),
    & env_ee(itech,ienvr+1),env_nt(itech,ienvr+1),
    & env_ni(itech,ienvr+1),
    & env_ne(itech,ienvr+1),env_nf(itech,ienvr+1),
    & env_g(itech,ienvr+1),env_w(itech,ienvr+1)
```

Step 14: **The facilities well cost is designed based on the gas throughput from the well. The file currently has 12,000 Mcf/day as a "maximum" throughput, which is the maximum achievable rate for any reservoir in the database, and will not be exceeded with the current data set. This structure allows for facilities well costs to be applied in steps if desired, so that the facilities costs could vary with different production rates. These values are taken from EIA's "Costs and Indices for Domestic Oil and Gas Field Equipment and Production Operations", August 1996.**

```
       read(i0,*)
       read(i0,*) nreg_faci(itech)
       Do ireg=1,nreg_faci(itech)
       read(i0,*)
       read(i0,*) faci_reg(itech,ireg),fac_n(itech,ireg)
       read(i0,*)
```

```
         read(i0,*)
         do istep=1,fac_n(itech,ireg)
          read(i0,*) faci_max(istep,itech,ireg),faci_k(istep,itech,ireg),
      @             faci_s(istep,itech,ireg)
         enddo
         Enddo
         read(i0,*)
         read(i0,*) faci_reg(itech,qreg+1),fac_n(itech,qreg+1)
         read(i0,*)
         read(i0,*)
         do istep=1,fac_n(itech,qreg+1)
          read(i0,*) faci_max(istep,itech,qreg+1),
      @             faci_k(istep,itech,qreg+1),
      @             faci_s(istep,itech,qreg+1)
         enddo
```

Step 15: **Stimulation Factor (*STMFAC*) is defined as design factor in calculating fracture cost. A value of 0.60 means that if the reservoir is fractured for a 500 ft fracture half length, then the actual cost would be for a fracture of 500/0.60 = 833.33 ft.**

```
        read(i0,*)
        read(i0,*) stimfac(itech)
```

Step 16: **The compression costs assume that single stage compressor is used.**

```
        read(i0,*)
        read(i0,*) cost_bhp(itech)
```

Step 17: **Variable O&M gas cost depends upon how much gas is handled and represents electricty use and cost factors such as more trips to the fields, etc. Most of the variable O&M is the compressor O&M. The fixed O&M values have also been calculated from the "Costs and Indices for Domestic Oil and Gas Field Equipment and Production Operations" August 1996 report in similar fashion to the facilities well cost. It is a function of well depth, and again, the regions must be set.**

```
        read(i0,*)
        read(i0,*) oam_h2o(itech)
        read(i0,*)
        read(i0,*) oam_gas(itech),oam_inc(itech)
        read(i0,*)
        read(i0,*) comp_vc(itech)
        read(i0,*)
        read(i0,*)
        read(i0,*) nreg_fx(itech)
        call chkdim(nreg_fx(itech),qreg,'qreg')
        do ireg=1,nreg_fx(itech)
         read(i0,*)
         read(i0,'(a,t5,i2)') fxoam_reg(itech,ireg),fxoam_n(itech,ireg)
         call chkdim(fxoam_n(itech,ireg),qstep,'qstep')
         read(i0,*)
         read(i0,*)
         do istep=1,fxoam_n(itech,ireg)
          read(i0,*) fxoam_max(istep,itech,ireg),
     &     fxoam_k(istep,itech,ireg),
     &     fxoam_s(istep,itech,ireg)
         enddo
        enddo
        read(i0,*)
        read(i0,'(a,t5,i2)')
     &    fxoam_reg(itech,qreg+1),fxoam_n(itech,qreg+1)
        call chkdim(fxoam_n(itech,qreg+1),qstep,'qstep')
        read(i0,*)
        read(i0,*)
        do istep=1,fxoam_n(itech,qreg+1)
         read(i0,*) fxoam_max(istep,itech,qreg+1),
     &    fxoam_k(istep,itech,qreg+1),
     &    fxoam_s(istep,itech,qreg+1)
        enddo
       enddo
 100   format(t3,f10.0)
 10    format(i3)
```

Step 18: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program RD_COST() is ended.**

```
       Return
       End
```

SUB-PROGRAM RD_GEO()

LOCATION: READINP.FOR

MAIN THEME: This routine reads pay grade geologic distribution of selected reservoir properties from input file GEOLOGY.DAT. Pay grade geologic distribution within a reservoir is assumed to be dependent on resource type. Typically, conventional and coal bed show different characteristics. Coal bed reservoirs show less heterogeneity compared to conventional reservoirs, and hence there is more variation in the conventional resource type. To study the sensitivity for homogeneous reservoirs, the values specified should be modified to 1.0.

The pay grade geologic distribution among reservoirs should ensure that the following equation holds good:

Area*Porosity*Netpay*H2O Saturation For Pay Grade 1 +
Area*Porosity*Netpay*H2O Saturation For Pay Grade 2 +
Area*Porosity*Netpay*H2O Saturation For Pay Grade 3 = 1.0

So, for example in the case of conventional resource the values could be assigned as follows:

$0.20*1.5 + 0.50*1.0 + 0.30*0.667 = 1.0$ and so on.

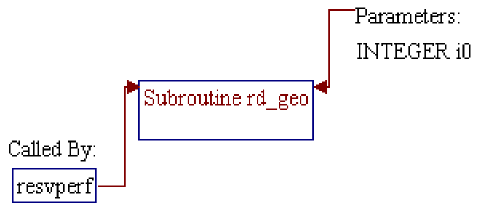
CALLS: None

CALLED BY: RESVPERF (in file RESVPERF.FOR)
Main program of Reservoir Performance Module.

READS: GEOLOGY.DAT
(Information on reservoir property distributions by pay grade)

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameter of the sub-program are declared. Header ".h" files are included. Local variable is declared.**

Note: Name of the sub-program is RD_GEO() and the parameter passed to this sub-program is as follows:

- *i0* Unit number for input file GEOLOGY.DAT

```
      subroutine rd_geo(i0)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'geology.h'
```

Note: Local variable is declared.

```
      integer i0
```

Step 2: **The reservoir property distribution among pay grades is specified below. This distribution is specified by resource type. So, all the reservoirs within a resource type are specified similar distribution. Variable *nrestype* is number of resource types for which pay grade property distribution is specified.**

```
      read(i0,*)
      read(i0,*) nrestype
```

Step 3: **If number of resource types (*nrestype*) for which pay grade distribution is specified is higher than maximum allowed (*qrestype=6*), then the RP module terminates after printing a fatal error message.**

```
      if(nrestype.gt.qrestype) then
       write(6,*) 'Number of reservoir type exceeds maximum allowed'
       write(6,*) 'Program must be Recompiled'
       stop
      endif
```

Step 4: **Data for pay grade property distribution is read. Area (*area_fac*), porosity (*por_fac*), net pay thickness (*netpay_fac*), initial water saturation (*h2osat_fac*) and permeability**

(*perm_fac*) values for each pay grade (3 in total for all the resource types) are assigned by resource type.

```
      read(i0,*)
      read(i0,*)
      do irestype=1,nrestype
       do ipay=1,3
        read(i0,*) res_map(irestype),id,area_fac(ipay,irestype),
     &    por_fac(ipay,irestype),netpay_fac(ipay,irestype),
     &    h2osat_fac(ipay,irestype),perm_fac(ipay,irestype)
       enddo
      enddo
```

Step 5: **The default pay grade geologic distribution is read in the following section. These distributions are applied to all the reservoirs for which distribution is not specified earlier. The default values are read into *qth+1* dimension for all the pay grades.**

```
      do ipay=1,3
       read(i0,*) id,id,area_fac(ipay,qrestype+1),
     &   por_fac(ipay,qrestype+1),netpay_fac(ipay,qrestype+1),
     &   h2osat_fac(ipay,qrestype+1),perm_fac(ipay,qrestype+1)
      enddo
```

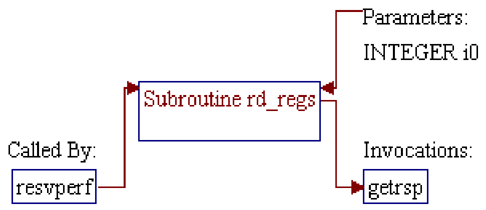
Step 6: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program RD_GEO() is ended.**

```
      Return
      End
```

SUB-PROGRAM RD_REGS()

LOCATION:	GSAM_A.FOR
MAIN THEME:	This routine reads REGIONS.DAT file which contains information about the list of the .GSM files to be run through the RP Module and several YES/NO switches as indicators for opening specific files for consistency checks.
CALLS:	GETRSP() (in file IOFUNCT.FOR) Transforms a yes/Yes or no/NO response to a logical true and false.
CALLED BY:	RESVPERF (in file RESVPERF.FOR) Main program of Reservoir Performance Module.
READS:	REGIONS.DAT (List of .GSM files to be run through RP Module)
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameter of the sub-program are declared. Header ".h" files are included. Local variables are declared.**

Note: Name of the sub-program is RD_REGS() and the parameter passed to this sub-program is as follows:

- *i0* Unit number for input file REGIONS.DAT

```
      subroutine rd_regs(i0)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'global.h'
```

Note: Local variables are declared.

```
      character*3 ch3
      logical getrsp
```

Step 2: **Reads number of years (*nyrset*) for which the reservoir performance module would run the type curves.**

```
      read(i0,*)
      read(i0,*) nyrset
      read(i0,*)
 809  format(t42,a3)
```

Step 3: **The following logical variables are used to print optional files from the reservoir performance module. These optional files are created for every GSAMID processed. The prefix is the last eight digits of GSAMID and the suffix are TCI (for type curve input), TCO (for type curve output) etc.**
l_tci **variable prints the type curve input files (*.TCI files)**
l_pro **variable prints the detailed pro-forma files (*.PRO files)**
l_tco **variable prints the type curve outout files (*.TCO files)**
l_prr **variable prints the reduced pro-forma entries (*.PRR file)**
l_npv **variable prints net present value summary (*.NPV files)**
prt_prs **variable prints bottomhole pressures in *.PRD files**

```
        read(i0,809) ch3
        l_tci=getrsp(ch3)
        read(i0,809) ch3
        l_pro=getrsp(ch3)
        read(i0,809) ch3
        l_tco=getrsp(ch3)
        read(i0,809) ch3
        l_prr=getrsp(ch3)
        read(i0,809) ch3
        l_npv=getrsp(ch3)
        read(i0,809) ch3
        prt_prs=getrsp(ch3)
```

Step 4: **The following section reads name of the .GSM file to be run through the reservoir performance module (*regnm(ireg)*), and the location of these files (*files(ireg)*). It also reads logical variable *ch3* which indicates whether type curve module has to be run or not. If *ch3* variable is set to "YES" then the type curve modules have to be run otherwise the type curve entries are read from (.BIN) files and economic calculations are performed.**

```
        ireg=1
        read(i0,*)
        read(i0,*)
150     read(i0,'(a,t10,a,t34,a)',end=160) regnm(ireg),files(ireg),ch3
        runtype(ireg)=getrsp(ch3)
        ireg=ireg+1
        goto 150
160     nreg=ireg-1
```

Step 5: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program RD_REGS() is ended.**

```
        Return
        End
```

SUB-PROGRAM RD_TAX()

LOCATION: READINP.FOR

MAIN THEME: This routine reads state income taxes, oil and gas severance taxes, and ad-voleram taxes as used in GSAM from input file TAXES.DAT. These numbers are taken from state publications (Chamber of Commerce) and from NEB (National Energy Board) publications for Canadian provinces. The values used are for Integrated oil and gas companies. It is realized that the tax structures for independent operators are little different. Once, the NRG data for "Dominant Operator Type" for the reservoir gets populated, the tax treatments could be changed.

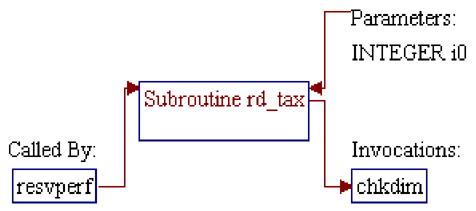
CALLS: CHKDIM() (in file IOFUNCT.FOR)
Checks if dimension of an array has been exceeded.

CALLED BY: RESVPERF (in file RESVPERF.FOR)
Main program of Reservoir Performance Module.

READS: TAXES.DAT
(Information about generic tax structure (capitalize versus expense switches) assumptions)

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameter of the sub-program are declared. Header ".h" files are included. Local variables are declared.**

Note: Name of the sub-program is RD_TAX() and the parameter passed to this sub-program is as follows:

- *i0* Unit number for input file TAXES.DAT

```
      subroutine rd_tax(i0)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'tax_reg.h'
```

Note: Local variables are declared.

```
      integer i0
      real*4  tstate,toil,tgas,toil_p,tgas_p
```

Step 2: **Reading various header lines and number of states (*ntax_st*) for which state income tax and severance tax are specified. Check is performed to ensure that dimensions are not exceeded.**

```
      read(i0,*)
      read(i0,*)
      read(i0,*) ntax_st
      call chkdim(ntax_st,qstate-1,'qstate')
      read(i0,*)
      read(i0,*)
```

Step 3: **The following do loop ensures that all tax information is read for the states. State income taxes (*strate*), oil severance taxes (*oil_sev*, *oil_sev_p*) , gas severance taxes (*gas_sev*, *gas_sev_p*), and ad-voleram (*advo*) taxes are read. Severance taxes are specified both as a percentage of revenue (e.g., *sev_rate*) and also as dollar per barrel or MCF of production (e.g. *gas_sev_p*). Default data is read which is used for reservoirs located in states for which data is not available.**

```
      do istate=1,ntax_st
       read(i0,*) tax_st(istate),tstate,toil,toil_p,tgas,tgas_p,advo
       strate(istate)=tstate/100.
```

```
        oil_sev(istate)=(toil+advo)/100.
        gas_sev(istate)=(tgas+advo)/100.
        gas_sev_p(istate)=tgas_p
        oil_sev_p(istate)=toil_p
       enddo
       read(i0,*) tax_st(qstate+1),tstate,toil,toil_p,tgas,tgas_p
       strate(qstate+1)=tstate/100.
       oil_sev(qstate+1)=toil/100.
       gas_sev(qstate+1)=tgas/100.
       gas_sev_p(qstate+1)=tgas_p
       oil_sev_p(qstate+1)=toil_p
```

Step 4: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program RD_TAX() is ended.**

```
       Return
       End
```

SUB-PROGRAM RD_TAX_NAT()

LOCATION: READINP.FOR

MAIN THEME: This routine reads TAX_NAT.DAT which contains information about generic tax structure (capitalize versus expense switches) assumptions.

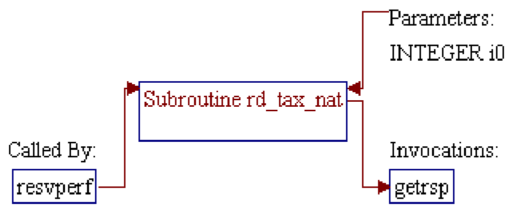
CALLS: GETRSP() (in file IOFUNCT.FOR)
Transforms a yes/Yes or no/NO response to a logical true and false.

CALLED BY: RESVPERF (in file RESVPERF.FOR)
Main program of Reservoir Performance Module.

READS: TAX_NAT.DAT
(Information about generic tax structure (capitalize versus expense switches) assumptions)

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameter of the sub-program are declared. Header ".h" files are included. Local variables are declared.**

Note: Name of the sub-program is RD_TAX_NAT() and the parameter passed to this sub-program is as follows:

- *i0* Unit number for input file TAX_NAT.DAT

```
      subroutine rd_tax_nat(i0)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'tax_nat.h'
      include 'tax_reg.h'
```

Note: Local variables are declared.

```
      integer i0
      character*3 resp
      logical getrsp
```

Step 2: **Read various entries from the TAX_NAT.DAT file. U.S. and Canadian federal income taxes are read first. They are converted into fraction by dividing by 100.**

```
      read(i0,*)
      read(i0,*) fedrate_us
      fedrate_us=fedrate_us/100
      read(i0,*)
      read(i0,*) fedrate_can
      fedrate_can=fedrate_can/100
```

Step 3: **Independent producer depletion rate (*ipdr*) is read and converted into fraction.**

```
      read(i0,*)
      read(i0,*)ipdr
      ipdr=ipdr/100
```

Step 4: **The following YES/NO responses are for expensing versus capitalizing. The following questions are for whether intangible drilling costs need to be capitalized, and whether other intangibles need to be capitalized.**

```
      read(i0,*)
      read(i0,10)resp
      cidc=getrsp(resp)
      read(i0,*)
      read(i0,10) resp
      coi=getrsp(resp)
```

Step 5: **The following YES/NO responses are for environmental costs. Variable *envscn* indicates whether environmental costs should be included in the cash flow calculations. The next question is whether environmental costs need to be capitalized.**

```
      read(i0,*)
      read(i0,10) resp
      envscn=getrsp(resp)
      read(i0,*)
      read(i0,10) resp
      ce=getrsp(resp)
```

Step 6: **The alternative minimum tax assignments are done in the following sections.**

```
      read(i0,*)
      read(i0,10)resp
      amt=getrsp(resp)
      read(i0,*)
      read(i0,10) resp
      credamt=getrsp(resp)
      read(i0,*)
      read(i0,*) smar
      smar=smar/100
```

Step 7: **Intangible drilling cost preference deductions, ACE rate, AMT tax rate etc., are assigned.**

```
      read(i0,*)
      read(i0,*)ipd
      ipd=ipd/100
      read(i0,*)
      read(i0,*)acer
      acer=acer/100
      read(i0,*)
      read(i0,*)ira
      ira=ira/100
      read(i0,*)
      read(i0,*)amtrate
      amtrate=amtrate/100
```

Step 8: **Expense Environmental Cost counter, net income limitation, net income limitation limit percentage, percent depletion rates,**

percent of intangible investment to capitalize, EOR tax credit rates are specified.

```
        read(i0,*)
        read(i0,10) resp
        eec=getrsp(resp)
        read(i0,*)
        read(i0,10) resp
        nil=getrsp(resp)
        read(i0,*)
        read(i0,*) nill
        nill=nill/100
        read(i0,*)
        read(i0,*)pdr
        pdr=pdr/100
        read(i0,*)
        read(i0,*)piic
        piic=piic/100
        read(i0,*)
        read(i0,*)eortcr
        eortcr=eortcr/100
```

Step 9: **Various flags and percentages for G&G and lease acquisition are assigned.**

```
        read(i0,*)
        read(i0,10) resp
        ggctc=getrsp(resp)
        read(i0,*)
        read(i0,*)ggctcr
        ggctcr=ggctcr/100
        read(i0,*)
        read(i0,10) resp
        ggetc=getrsp(resp)
        read(i0,*)
        read(i0,*)ggetcr
        ggetcr=ggetcr/100
        read(i0,*)
        read(i0,10) resp
        lactc=getrsp(resp)
        read(i0,*)
        read(i0,*)lactcr
        lactcr=lactcr/100
        read(i0,*)
        read(i0,10) resp
        laetc=getrsp(resp)
        read(i0,*)
        read(i0,*)laetcr
        laetcr=laetcr/100
```

Step 10: **Tangible, intangible development drilling and other intangible tax credit flags, counters and variables are set.**

```
        read(i0,*)
        read(i0,10) resp
        tdtc=getrsp( resp)
        read(i0,*)
        read(i0,*)tdtcr
        tdtcr=tdtcr/100
        read(i0,*)
```

```
      read(i0,10) resp
      idctc=getrsp(resp)
      read(i0,*)
      read(i0,*) idctcr
      idctcr=idctcr/100
      read(i0,*)
      read(i0,10) resp
      oitc=getrsp(resp)
      read(i0,*)
      read(i0,*)oitcr
      oitcr=oitcr/100
```

Step 11: **Environmental cost tangible, intangible and operating cost tax credit flags, counters and values are assigned.**

```
      read(i0,*)
      read(i0,10) resp
      ettc=getrsp(resp)
      read(i0,*)
      read(i0,*) ettcr
      ettcr=ettcr/100
      read(i0,*)
      read(i0,10) resp
      eitc=getrsp(resp)
      read(i0,*)
      read(i0,*)eitcr
      eitcr=eitcr/100
      read(i0,*)
      read(i0,10) resp
      eoctc=getrsp(resp)
      read(i0,*)
      read(i0,*)eoctcr
      eoctcr=eoctcr/100
```

Step 12: **Tax credit flags and values on tangible and intangible investments are assigned.**

```
      read(i0,*)
      read(i0,10) resp
      tcoti=getrsp(resp)
      read(i0,*)
      read(i0,*)yr1
      read(i0,*)
      read(i0,10) resp
      tcoii=getrsp(resp)
      read(i0,*)
      read(i0,*)yr2
 10   format(a)
```

Step 13: **Percent of G&G that could be depleted is specified and flag for forgiveness of state taxes is assigned. Number of years for forgiveness of state taxes is also specified.**

```
      read(i0,*)
      read(i0,*)pggc
      pggc=pggc/100
      read(i0,*)
      read(i0,10) resp
      fsttax=getrsp(resp)
      read(i0,*)
      read(i0,*)yr3
```

Step 14: **Percent of lease acquisition cost that could be capitalized in a year is specified.**

```
      read(i0,*)
      read(i0,*)plac
      plac=plac/100
```

Step 15: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program RD_TAX_NAT() is ended.**

```
      Return
      End
```

SUB-PROGRAM RD_TECH()

LOCATION: READINP.FOR

MAIN THEME: This routine reads TECH.DAT which contains information on number of technologies and data specifications for each technology.

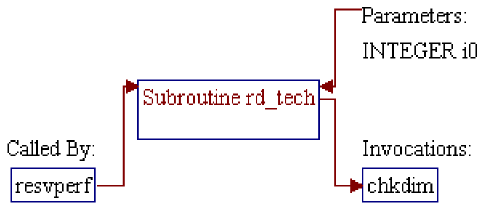
CALLS: CHKDIM() (in file IOFUNCT.FOR)
Checks if dimension of an array has been exceeded.

CALLED BY: RESVPERF (in file RESVPERF.FOR)
Main program of Reservoir Performance Module.

READS: TECH.DAT
(Information on number of technologies and data specifications for each technology)

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameter of the sub-program are declared. Header ".h" files are included. Local variables are declared.**

Note: Name of the sub-program is RD_TECH() and the parameter passed to this sub-program is as follows:

- *i0* Unit number for input file TECH.DAT

```
      subroutine rd_tech(i0)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'cost.h'
      include 'gsamvar.h'
      include 'tech.h'
```

Note: Local variables are declared.

```
      integer i0,npreg,npres,npay,nmod
```

Step 2: **Number of technologies (*ntech*) are specified and dimension is checked.**

```
      read(i0,*)
      read(i0,*) ntech
      call chkdim(ntech,qtech,'qtech')
```

Step 3: **Do loop starts for the number of technologies. Technology data is always read for all the technologies specified. In most of cases, it is read for both current technology and advanced technology.**

```
      do itech=1,ntech
```

Step 4: **For all the regions available in the RP module, the well type is initialized as vertical (i.e. *jtyp_tech=0*).**

```
       do ireg=1,qreg
        jtyp_tech(itech,ireg)=0
       enddo
```

Step 5: **Name of the technology (*technm*) is specified.**

```
      read(i0,*)
      read(i0,993) technm(itech)
993   format(a20)
```

Step 6: **In the following section, development well dry hole rate (*prob_dry*) is read. In this case it is 20%; i.e. if 100 wells are drilled in a reservoir then 20 of them are dry wells and the rest 80 are successful development wells. This provides a good estimate on the number of total wells drilled and the costs associated. The development well success rates are specified later in the E&P module as a function of play (file PLY_DFN.SPC). Hence, the value of 20% of the RP module is overwritten by the E&P module value.**

```
      read(i0,*)
      read(i0,*) prob_dry(itech)
      prob_dry(itech)=prob_dry(itech)/100
```

Step 7: **The type curve for water drive reservoir is designed in such a manner that it needs the year (from start of production) when first infill drilling is performed. This value is generally assumed to be 5 years. For all other resource types, the type curve computes the year for infill driling (which is the year in which the primary wells can't sustain the constant production rate).**

```
      read(i0,*)
      read(i0,*) wdtim_tech(itech)
```

Step 8: **Number of regions (*npreg*) for which proration rule is defined is specified in the following section. It is checked and ensured that dimensions are not exceeded.**

```
      read(i0,*)
      read(i0,*) npreg
      call chkdim(npreg,qstate-1,'qstate')
```

Step 9: **Proration rates are defined by region and/or by state (for Appalachia). Generally, it is assumed that the horizontal wells could produce at a faster rate than the vertical wells. It is also assumed that on average the horizontal wells could produce at**

25% to 30% higher rates compared to the vertical wells. This is based on various SPE papers and the horizontal well technology text books. Proration data for different regions and the default value are read.

Advanced technology improves the proration anywhere from 10 percent to 40 percent. These values could be changed. We have assumed a lower improvement in the low productive regions, and moderate improvement in high productive regions. In The Gulf of Mexico, the increase is not much (0.45 versus 0.40 in current technology) because, current practices itself is very effective and there is a very small improvement that could be achieved by utilizing advanced technology. The production rates in Gulf is more constrained by pipeline capacities and not by reservoir producability.

```
      read(i0,*)
      do ireg=1,npreg
       read(i0,*)irxx,prorat_tech(itech,irxx)
      enddo
      read(i0,*)
      read(i0,*)ixxx,prorat_tech(itech,qstate)
```

Step 10: **In case proration factors are specified by state then the following section is invoked and data is read.**

```
      read(i0,*)
      read(i0,*) ntech_st
      call chkdim(ntech_st,qstate-1,'qstate')
      read(i0,*)
      do istate=1,ntech_st
       read(i0,*) tech_st(istate),proration(itech,istate)
      enddo
```

Step 11: **Pay enhancement factor is read in the following section. It relates fraction of the pay zone which was not drained in primary drilling. This value is primarily used for infill drilling and changes the percent of pay zone contacted. For coal bed it is assumed that there is no pay enhancement by infill drilling. It is assumed that use of current technology doesn't improve the pay continuity.**

```
      read(i0,*)
      read(i0,*) npay
      call chkdim(npay,qreg,'qreg')
      read(i0,*)
      do ireg=1,npay
       read(i0,*)irxx,pay_tech(itech,irxx)
      enddo
      read(i0,*)
```

```
        read(i0,*)ixxx,pay_tech(itech,qreg)
```

Step 12: **Minimum system pressure (wellhead pressure) is specified in the following section. It is assumed that for the Appalachia region the gas reservoir could be produced when the system pressure reaches very close to atmospheric pressure. The information was obtained from various gas operators in Appalachian basin. System pressures are read for all the regions specified. In addition, default value for system pressure is also read which is used for reservoirs for which data is not specified in the regional average.**

```
        read(i0,*)
        read(i0,*) npres
        call chkdim(npres,qreg,'qreg')
        read(i0,*)
        do ireg=1,npres
         read(i0,*)irxx,psys_tech(itech,irxx)
        enddo
        read(i0,*)
        read(i0,*)ixxx,psys_tech(itech,qreg)
```

Step 13: **The following section reads skin factors (*fracsk_tech*) for vertical wells by resource type. The values specified are total skin factors. This includes the skin related to Non-Darcy flow, Completion technology, and other factors that increase the pressure drop around the wellbore. These skin factors are for vertical wells. For analyzing horizontal well behavior, the vertical well skin factors are used to calculate equivalent horizontal well skin based on vertical permeability, horizontal permeability, net pay and horizontal well length.**

```
        read(i0,*)
        read(i0,*)nmod
        call chkdim(nmod,qrestype,'qrestype')
        read(i0,*)
        read(i0,*)(fracsk_tech(itech,imod),imod=1,nmod)
```

Step 14: **Variable *wrad_tech* is well radius in ft. In most of the case, a 9 inch hole is assumed for all the wells operating in the reservoir. Well radius is specified all the resource types available.**

```
        read(i0,*)
        read(i0,*)(wrad_tech(itech,imod),imod=1,nmod)
```

Step 15: **Fracture half length (*fracxf_tech*) is specified for all the resource types available.**

```
        read(i0,*)
        read(i0,*)(fracxf_tech(itech,imod),imod=1,nmod)
```

Step 16: **Fracture conductivity (*fraccn_tech*) is specified for all the resource types available.**

```
        read(i0,*)
        read(i0,*)(fraccn_tech(itech,imod),imod=1,nmod)
```

Step 17: **The following section sets up horizontal well data. A value of 1 for variable *jtyp_tech* indicates that all wells in the reservoir are horizontal. The next variable *jlen_tech* is horizontal well length.**

```
       read(i0,*)
       read(i0,*)njreg
       call chkdim(njreg,qreg,'qreg')
       read(i0,*)
       do ireg=1,njreg
       read(i0,*)irxx,jtyp_tech(itech,irxx),jlen_tech(itech,irxx)
       enddo
```

Step 18: **In the following section, tubing diameter is assigned. The vertical wells situated in Appalachia are assumed to be draining from a 2 7/8" (diameter) tubing; which means 1.4 inches of tubing radius for Appalachia. For all other regions a 4 inch diameter tubing is used for vertical wells. The horizontal wells on the other hand, are assumed to be bigger in diameter (9 3/8 inches).**

```
       read(i0,*)
       read(i0,*)ndreg
       call chkdim(ndreg,qreg,'qreg')
       read(i0,*)
       do ireg=1,ndreg
       read(i0,*)irxx,diam_tech(itech,irxx)
       enddo
       read(i0,*)
       read(i0,*)irxx,diam_tech(itech,qreg)
       read(i0,*)
```

Step 19: **The technology loop is closed and the program control is returned back to the calling routine (program RESVPERF) and the sub-program RD_TECH() is ended.**

```
        Return
        End
```

SUB-PROGRAM RD_TEMP()

LOCATION: GSAM_A.FOR

MAIN THEME: This routine reads a type curve input file (TEMPLATE.DAT file) and store lines in variable *lines()*. It is used in creating *.TCI file.

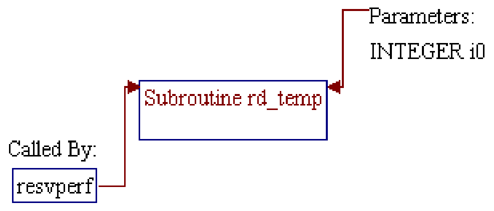
CALLS: None

CALLED BY: RESVPERF (in file RESVPERF.FOR)
Main program of Reservoir Performance Module.

READS: TEMPLATE.DAT
(Template file for type curve input file .TCI)

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameter of the sub-program are declared. Header ".h" files are included. Additional common block and local variables are declared.**

Note: Name of the sub-program is RD_TEMP() and the parameter passed to this sub-program is as follows:

- *i0* Unit number for input file TEMPLATE.DAT

```
      subroutine rd_temp(i0)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'gsamvar.h'
```

Note: Additional common block and local variables are declared.

```
      integer iline,i0
      character*80 lines(qline)
      common/ddd/lines
```

Step 2: **The following section reads the entries from TEMPLATE.DAT file and stores them in *lines()* variable. These entries are later written in *.TCI file for checking database entries by pay grade.**

```
      iline=1
 10   read(i0,'(a80)',end=20) lines(iline)
       iline=iline+1
       if(iline.gt.qline) stop 4092
       goto 10
 20   continue
```

Step 3: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program RD_TEMP() is ended.**

```
      Return
      End
```

SUB-PROGRAM RD_UND()

LOCATION: READONE.FOR

MAIN THEME: This routine reads the undiscovered reservoir database file, i.e., the [UND*].GSM file. These [GSAM].GSM files are created from GSAM Resource Module. Data sources are USGS, MMS, EIA, OGJ and other ICF internal estimates. Each record in the [UND*].GSM file indicates a field size class in a play denoted by a GSAMID. Number of undiscovered reservoir accumulations are read for the corresponding GSAMID. The main program in RP module, RESVPERF, calls RD_UND() for every GSAMID and collects all the data through common blocks.

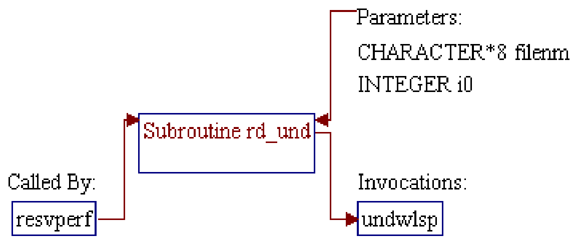
CALLS: UNDWLSP() (in file READONE.FOR)
Calculates well spacing for undiscovered resource of GSAM.

CALLED BY: RESVPERF (in file RESVPERF.FOR)
Main program of Reservoir Performance Module.

READS: [UND*].GSM
(Undiscovered reservoir database files (1 line format))

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program RD_UND are declared. Header ".h" files are included and local variables and common blocks are defined.**

Note: Name of the sub-program is RD_UND() and the parameters passed to this sub-program are as follows:

• *filenm*	Name of the [UND*].GSM file for which data is read.
• *i0*	Unit number of the *.GSM file which gets read in the routine.
• *	In case "end of file" is encountered the "*" indicates that the control has to open another [GSAM].GSM file; or in case all reservoirs in all the available [GSAM].GSM files are read, then the RP module has to terminate.

```
            subroutine rd_und(filenm,i0,*)
```

Note: All the variables are stored in the header files as shown below. In addition, local variable filenm is also defined.

```
          include 'gsamvar.h'
          include 'dimen.h'
          include 'tech.h'
          include 'global.h'
          character*8  filenm
```

Note: Heating value of the natural gas is hard wired to 909.1 BTU/cubic ft. In addition, impurity levels are initialized to zero (*h2s*, *co2* and *n2* values are set to zero).

```
          heatvl = 909.1
101       h2s = 0.0
          co2 = 0.0
          n2 = 0.0'
```

Step 2: **Sequential READING of entries for a GSAMID (for undiscovered resource GSAMID indicates a field size class in a play) starts.**

Note: Reading is continued using the following read statement. As shown below this read statement is formatted and the format statement is specified in statement number 1. Majority of the variable names are self-explanatory. Variable *resno* indicates the number of undiscovered reservoir accumulation in the field size class of the play (indicated by the appropriate GSAMID). *istartappl* variable is used for APPL.GSM for which this variable indicates the first year the reservoir started to produce. *h20dep* is the water depth in case of an offshore reservoir. The EIA code for the corresponding undiscovered GSAMID is set to *'00000000'* and if *gasgrv* value is missing in the [GSAM].GSM file, it is hard wired to *0.65*.

```
           READ(i0,1,end=201) gsamid,acprod,netpay,perm,por,watsat,depth
     +     ,presin,bhtemp,co2,n2,h2s,resno,state,gasgrv,istartappl,
     +      h2odep
 1         format(a11,t15,f7.0,1x,f5.1,1x,f8.3,1x,f5.3,1x,f5.3,1x,
     +      f6.0,1x,f6.0,1x,f4.0,1x,f6.4,1x,f6.4,1x,f6.4,t92,
     +      f9.0,1x,i6,1x,f5.3,1x,i4,t139,f6.0)
            eiacod = '00000000'
            if (gasgrv.le.0.0) gasgrv = 0.65
```

Step 3: **Various assignments are performed. Permeability value is assigned to variable *perhor* and vertical permeability is assumed to be 30% of horizontal permeability. Matrix permeability is assumed to be 10% of horizontal permeability. Matrix porosity is assigned to be maximum of either 4% or 5% less than the total porosity of the reservoir rock. Gross pay is assumed to be twice the net pay. Well inside radius is assumed to be 4.248 inches (0.354 ft). For wells in Appalachia it is assumed to be 2.4 inches (0.20 ft). Formation compressibility for all undiscovered reservoirs is assumed to be 3x10^{-6} psi^{-1}. Aquifer permeability is assumed to be same as the horizontal permeability of the reservoir. Induced fracture skin factor (*fracsk*) and fracture half length (*fracxf*) is initialized. In addition, the fracture conductivity (*fraccn*) is assumed to be 50,000 md-ft for all reservoirs.**

```
       perhor = perm
       pervrt = perm*0.30
       permtx = 0.1*perm
       portot = por
       pormtx = max(0.04,portot-0.05)
       porcur = por
       grspay = 2.0*netpay
       welrad = 0.354
       if (filenm(1:3).eq.'APP'.or.filenm(1:3).eq.'app') welrad=0.20
       compfr = 3.0*10e-6
       aquprm = perm
       fracsk = 0.0
       fracxf = 0.0
       fraccn = 50000.0
       gassat = 1.0 - watsat
```

Step 4: **The read statements below assigns GSAM supply region (*gsamsr*), reservoir status (*statin*), and module counter (*module*)**

from variable GSAMID. For module types 2, 3, and 4 the following fracture properties are also assigned. Remember, these entries would get overwritten from the entries in TECH.DAT file if they are available. Natural fracture spacing is assigned to be 0.2 ft, and fracture skin is assigned a value of +2.0 Induced fracture conductivity is assigned a value of 5000 md-ft for all fractures and fracture half length is set to 400 ft. For Appalachia, *fracxf* is assigned to be 150 ft.

```
read(gsamid(1:2),'(i2)') gsamsr
read(gsamid(3:3),'(i1)') statin
read(gsamid(4:4),'(i1)') module
if(module.gt.1.and.module.le.4)then
 fracsp = 0.2
 fracsk = 2.0
 if (fraccn.lt.1) fraccn = 5000.0
 if(fracxf.lt.100.) fracxf=400.
 if(gsamsr.eq.1) fracxf=150.
endif
```

Step 5: **Well spacing for undiscovered reservoirs are assigned based on the sub-program UNDWLSP(). Well spacing for undiscovered reservoirs are primarily assigned based on depth and resource type.**

```
wlspac=undwlsp(gsamsr,module,depth,acprod,state,filenm,gsamid)
```

Step 6: **Drive type is assigned based on module type. For modules 1 and 2, drive type is assigned to 1 indicating pressure depletion. For water drive reservoirs, the drive type is 2 indicating water drive. Finally, for coal bed methane reservoirs, the drive type is assigned to 3 indicating desorption/depletion. The field size class (9, 10 and 11 digits in GSAMID) is assigned (*rescod*).**

```
IF (module.EQ.1) drive = 1
IF (module.EQ.2) drive = 1
IF (module.EQ.5) drive = 2
IF (module.EQ.6) drive = 3
read(gsamid(9:11),'(i3)') rescod
```

Step 7: **The following section of the code is for coal bed methane. In case, the reservoir is not a coal bed methane (module not equal to 6) then the control skips the calculations and returns to the main calling routine.**
The default nature of coal bed methane (U.S. and Canada) is dry (*iunctype = 0*). For coal bed reservoirs in Canada, the gas content is assumed to be 400 SCF/TON and sorption time is assumed to be 30 days.

If the water saturation in the reservoir is higher than 90% it is assumed to be wet coal (*iunctype* = *1*). The variable *iunctype* = *2* is for dry shale and *iunctype* = *3* for wet shale.
The default location for coal bed methane is western coal (*iuncloc* = *2*).
For Appalachia (*gsamsr* = *1*), the location variable *iuncloc* is set to 0 and the coal content is hardwired to 200 SCF/TON.
For MAFLA (*gsamsr = 2*), *iunctype* is set to 1 indicating wet coal, and *iuncloc* is set to 1.
For Midwest (*gsamsr = 3*), *iunctype* is set to 3 indicating wet shale, and *inncloc* is set to 1. The shale content is hardwired to 50 SCF/TON.
For San Juan (*gsamsr = 9*), *iunctype* is set to 1 indicating wet coal, and *iuncloc* is set to 2 indicating western coal.
Finally, reservoir type (*irestype*) variable is assigned the value of *module.*

```
      if (module.ne.6) goto 200
      iunctype=0
      if (gsamsr.ge.22) then
       gascon = 400.0
       srptim = 30.
      endif
      if (watsat.ge.0.90) then
        iunctype = 1
      endif
      iuncloc = 2
      if(gsamsr.eq.1) then
       iuncloc=0
       gascon = 200.0
      elseif (gsamsr.eq.2) then
       iuncloc=1
       iunctype=1
      elseif (gsamsr.eq.3) then
       iuncloc=1
       iunctype=3
       gascon = 50.0
      elseif (gsamsr.eq.9) then
       iuncloc=2
       iunctype=1
      endif
      irestype=module
```

Step 8: **At this stage, the routine terminates. The *"RETURN"* associated with statement number *200* indicates that the control goes to the main calling routine and once processing for the current GSAMID is complete, next record is read.**
The *"RETURN"* associated with statement number *201* indicates that "End of File" has been encountered and next [UND*].GSM file needs to be opened and read OR the RP Module terminates because all reservoirs have been processed.

```
  200  return
  201  return 1
```

```
        end
```

SUB-PROGRAM READONE()

LOCATION: READONE.FOR

MAIN THEME: This routine reads the discovered reservoir database file, i.e., the [GSAM].GSM file. These [GSAM].GSM files are created from NRG datasets for U.S. and Canada. There are 13 records (lines) per GSAMID (i.e. reservoir). The main program in RP module, RESVPERF, calls READONE() for every GSAMID and collects all the data through common blocks. In case of error in reading a record from the file, the GSAMID is reported in an [GSAM].ERR file and next GSAMID is read. The NRG U.S. database contains latest production data upto year 1993 and for Canada it is upto 1994. This distinction is made while reading the [GSAM].GSM file.

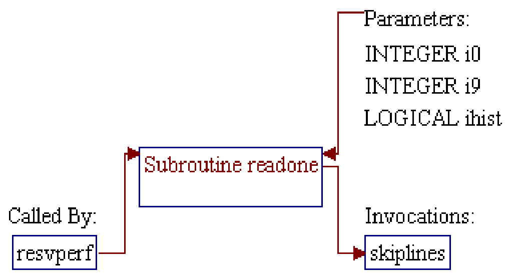
CALLS: SKIPLINES() (in file READONE.FOR)
The SKIPLINES routine skips the required number of lines which should be skipped before next GSAMID can be read. This occurs whenever there is an error in the database entry for a particular GSAMID and the reservoir can not be processed in a normal GSAM run.

CALLED BY: RESVPERF (in file RESVPERF.FOR)
Main program of Reservoir Performance Module.

READS: [GSAM].GSM
(Discovered reservoir database files)

CREATES: [GSAM].ERR
(Error messages)

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program READONE() are declared. Header ".h" files are included and local variables and common blocks are defined.**

Note: Name of the sub-program is READONE() and the parameters passed to this sub-program are as follows:

•	*i9*	unit number of the [GSAM].ERR file which contains the GSAMID's which can not be processed through the type curves.
•	*i0*	unit number of the [GSAM].GSM file which gets read in the routine.
•	*ihist*	TRUE/FALSE toggle for history check calculations. If this toggle is set to TRUE (indicating response "YES") history check is performed for the existing producing reservoirs.
•	*	In case "end of file" is encountered the "*" indicates that the control has to open another [GSAM].GSM file; or in case all reservoirs in all the available [GSAM].GSM files are read, then the RP module has to terminate.

```
      subroutine readone(i9,i0,ihist,*)
```

Note: All the variables are stored in the header file GSAMVAR.H.

```
    include 'gsamvar.h'
```

Note: Local variables are defined and logical counter w_line is set to *FALSE*. If this counter is set to *TRUE* write statements gets activated which prints on the screen the line numbers on which the control is currently active.

```
     integer i9
     logical w_line,ihist
     w_line=.false.
```

Step 2: **Sequential READING of entries for a GSAMID (reservoir) starts.**

Note: Statement #234 indicates that the program control comes to read yet another reservoir entry (i.e. 13 lines) in case there is an error in the current reservoir. Since, *w_line* variable is set to *FALSE* the write statements never gets activated.

```
 234     if(w_line) write(6,*) 'reading line 1'
```

Step 3: **The first two lines of the [GSAM].GSM file are read. Data reading is performed through the [GSAM].GSM file *(i0)* and format statement (1001 or 1002). When normal end of file is encountered the program control goes to 200 which brings the control back to the main program RESVPERF. In case of error, the control goes to the corresponding error statement line as shown below.**

Note: The read statements for the first two lines of the 13 line discovered reservoir [GSAM].GSM file are formatted as shown below. The first line is read through a format statement 1001 and second line by 1002. In the first record, GSAMID, reservoir code and name, field name, 4 digit state code, play code, field size class etc. are read. In the second line, township, onshore/offshore counter, GSAM supply region, status of the reservoir, latitude, longitude, depth, water depth etc. are read and the variables are stored in GSAMVAR.H.

```
         read(i0,1001,end=200,err=201)
      &  gsamid,rescod,fldnam,eiacod,rsvnam,state,county,plycod,bsncod,
      &  fsclas
1001     format(a11,1x,i8,1x,a30,1x,a10,1x,a30,1x,i4,1x,i5,1x,i5,1x,
      &       i3,1x,i3)
         if(w_line) write(6,*) 'reading line 2'
         read(i0,1002,end=200,err=202)
      &  twnshp,range,onoffs,gsamsr,statin,lat,lon,
      &  depcls,restype,trapty,drive,depth,deptss,h2odep
1002     format(a8,1x,a8,1x,i2,1x,i4,1x,i2,1x,f7.3,1x,f7.3,
      &       1x,i4,1x,i4,1x,i2,1x,i2,1x,f7.0,f7.0,f7.0)
```

Step 4: **Entry #3 through 13 in a [GSAM].GSM file for a GSAMID are unformatted read as shown below. In the following section, gross and net pay, drainage area, horizontal, vertical and matrix permeability, total and matrix porosity, initial water and gas saturation are read. In case, the initial gas saturation is missing in the [GSAM].GSM file, it is calculated from the value of water saturation.**

```
         if(w_line) write(6,*) 'reading line 3'
         read(i0,*,end=200,err=203)
      &  grspay,netpay,paydsp,weldrn,perhor,
      &  pervrt,permtx,portot,pormtx,porcur,watsat,gassat
         if (gassat.le.0.0) gassat = 1.0 – watsat
```

Step 5: **In record #4 of the [GSAM].GSM file, initial pressure, gravity, impurity levels, coal bed methane location and properties etc. are read and stored. Record #5 contains various fracture related data, discovery year of the field, and well spacing for the reservoir. In case of Canada (GSAM supply regions 22, 23 and 24), for discovered and undeveloped reservoirs (i.e. *status=2*, CANDU.GSM) the well spacing is hardwired to 80 acres.**

```
      if (w_line) write(6,*) 'reading line 4'
      read(i0,*,end=200,err=204)
   &  presin,gasgrv,bhtemp,heatvl,co2,n2,h2s,welrad,compfr,solgas,
   &  chlcon,cmpwat,inporf,langvl,langpr,prsdsp,gascon,iunctype,
   &  iuncloc
      if(w_line) write(6,*) 'reading line 5'
      read(i0,*,end=200,err=205)
   &  srptim,aqurad,aquprm,fracsp,fracwi,fracfl,fracxf,fraccn,
   &  fracsk,fracpo,watsaf,discyr,disfld,dismth,sgdvyr,domopr,wlspac
      if (gsamid(1:2).eq.'22'.or.gsamid(1:2).eq.'23'.
   &   or.gsamid(1:2).eq.'24') then
       if (gsamid(3:3).eq.'2') then
         wlspac = 80.0
       endif
      endif
```

Step 6: **Acreage, maximum proved area, OGIP, cumulative gas produced till 1993, gross reserves reported as of 1993, current reported pressure, proration rule (if available), NGL production (Barrels NGL/MMCF dry gas) etc. are read in the following section.**

```
     if(w_line) write(6,*) 'reading line 6'
     read(i0,*,end=200,err=206)
   & acprod,acprov,acprvd,acdv93,ogip,cgpr93,grsv93,rsvcls,
   & gwr93,prscur,watsac,watsab,prsflw,prorat,nglfact
```

Step 7: **Reservoir gas production rate (BCF/Year, pure methane production) is read for every year from 1982 to 1993 for reservoirs located in the U.S. For Canada, production data up to year 1994 is available and read. Once, the 1994 gas production rate for Canada is read, it is stored in variable *gasprd93* for history check calculations and other processing.**

```
     if(w_line) write(6,*) 'reading line 7'
     if (gsamid(1:2).eq.'22'.or.gsamid(1:2).eq.'23'.
   @  or.gsamid(1:2).eq.'24') then
      read(i0,*,end=200,err=207)
   &  gasprd82,gasprd83,gasprd84,gasprd85,gasprd86,gasprd87,
   &  gasprd88,gasprd89,gasprd90,gasprd91,gasprd92,gasprd93,gasprd94
      gasprd93 = gasprd94
```

```
        else
         read(i0,*,end=200,err=207)
      &  gasprd82,gasprd83,gasprd84,gasprd85,gasprd86,gasprd87,
      &  gasprd88,gasprd89,gasprd90,gasprd91,gasprd92,gasprd93
        endif
```

Step 8: **In the following sections of the [GSAM].GSM file, oil production rate, NGL production rate, total wells, gas producing wells, and shut-in wells are read for all the years from 1982 to 1993.**

```
       if(w_line) write(6,*) 'reading line 8'
         read(i0,*,end=200,err=208)
      &  oilprd82,oilprd83,oilprd84,oilprd85,oilprd86,oilprd87,
      &  oilprd88,oilprd89,oilprd90,oilprd91,oilprd92,oilprd93
       if(w_line) write(6,*) 'reading line 9'
       read(i0,*,end=200,err=209)
      &  nglprd82,nglprd83,nglprd84,nglprd85,nglprd86,nglprd87,
      &  nglprd88,nglprd89,nglprd90,nglprd91,nglprd92,nglprd93
       if(w_line) write(6,*) 'reading line 10'
       read(i0,*,end=200,err=210)
      &  totwel82,totwel83,totwel84,totwel85,totwel86,totwel87,
      &  totwel88,totwel89,totwel90,totwel91,totwel92,totwel93
       if(w_line) write(6,*) 'reading line 11'
       read(i0,*,end=200,err=211)
      &  prdwel82,prdwel83,prdwel84,prdwel85,prdwel86,prdwel87,
      &  prdwel88,prdwel89,prdwel90,prdwel91,prdwel92,prdwel93
       if(w_line) write(6,*) 'reading line 12'
       read(i0,*,end=200,err=212)
      &  shutwel82,shutwel83,shutwel84,shutwel85,shutwel86,
      &  shutwel87,
      &  shutwel88,shutwel89,shutwel90,shutwel91,shutwel92,shutwel93
```

Step 9: **In Record #13 of the [GSAM].GSM file, target well spacing, back pressure exponent, operating system back pressure, *P/Z* slope, field type and the type curve module to be used for the GSAMID are read for Canadian reservoirs. For the U.S. reservoirs in addition to the entries listed above, the counter indicating whether the reservoir is on federal or private land is also included. For the federal reservoir this last counter is set to 1 and for private it is set to 0.**

```
        if(w_line) write(6,*) 'reading line 13'
        if (gsamid(1:2).eq.'22'.or.gsamid(1:2).eq.'23'.or.
      &    gsamid(1:2).eq.'24') then
         read(i0,*,end=200,err=213)
      &   twlspac,bpslop,prssys,pzslop,fldtype,module
        else
          read(i0,*,end=200,err=213)
      &    twlspac,bpslop,prssys,pzslop,fldtype,module,frac_fed
        endif
```

Step 10: **For reservoirs (with history check flag set to "YES"), with latest reported gas (methane) production rate (*gasprd93*) less**

than 0.001 BCF/Year (i.e. approximately 3 MCF/Day) no calculations are performed and the reservoir is skipped completely before going to the type curve routines. The program control goes to statement number 234 and it reads the next reservoir entry.

```
       if (ihist.and.gasprd93.le.0.001) then
         goto 234
       endif
```

Step 11: **Some checks are performed. First, if the current reported reservoir pressure is greater than the initial reservoir pressure, the reservoir is skipped and next reservoir is read. Second, for module types 2, 3, and 4 (i.e. tight and fractured reservoirs) natural fracture spacing is set to 0.2 feet and induced fracture skin is set to +2.0. In addition, if induced fracture conductivity is missing (less than 1.0) then it is assigned to 50,000 md-ft, and if fracture half length is less than 100 ft it is assigned to 400 ft.**

```
       if (prscur.ge.presin) goto 234
       if(module.gt.1.and.module.le.4)then
         fracsp = 0.2
         fracsk = 2.0
         if (fraccn.lt.1)fraccn = 50000.0
         if(fracxf.lt.100.)fracxf=400.
       endif
```

Step 12: **For a reservoir which is not coal bed methane, the program control goes to 199 and the routines returns to the main calling program (RESVPERF).**

Note: For coal bed methane reservoirs, the following assignments are made. The default nature of coal bed methane is dry (*iunctype=0*). If the water saturation in the reservoir is higher than 90% it is assigned wet coal (*iunctype=1*). The variable *iunctype=2* for dry shale and *iunctype=3* for wet shale. The default location for coal bed methane is western coal (*iuncloc=2*). For Appalachia (*gsamsr=1*), the location variable *iuncloc* is set to 0 and the coal content is hardwired to 200 SCF/TON. For MAFLA (*gsamsr=2*), *iunctype* is set to 1 indicating wet coal, and *iuncloc* is set to 1. For Midwest (*gsamsr=3*), *iunctype* is set to 3 indicating wet shale, and *inncloc* is set to 1. The shale content is hardwired to 50 SCF/TON. For San Juan (*gsamsr=9*), *iunctype* is set to 1 indicating wet coal, and *iuncloc* is set to 2 indicating western coal.

```
       if (module.ne.6) goto 199
```

```
      iunctype=0
      if (watsat.ge.0.90) then
        iunctype = 1
      endif
      iuncloc = 2
      if (gsamsr.eq.1) then
       iuncloc=0
       gascon = 200.0
      elseif (gsamsr.eq.2) then
       iuncloc=1
       iunctype=1
      elseif (gsamsr.eq.3) then
       iuncloc=1
       iunctype=3
       gascon = 50.0
      elseif (gsamsr.eq.9) then
       iuncloc=2
       iunctype=1
      endif
```

Step 13: **If there are no errors in reading the entries from [GSAM].GSM file, the program terminates and the control goes to the main calling program (RESVPERF). In case there are errors, then the error is written in the [GSAM].ERR file and the remaining lines for the reservoir are skipped (by invoking SKIPLINES() routine). After all the remaining lines for the problem reservoir are read, the control goes to statement number 234 and next reservoir is read before going to the main calling routine. Finally, when all the entries in a GSM file are read, then the program terminates.**

```
199   return
200   continue
      return 1
 201  read(i0,*)
      write(i9,2) 'Error in Line 1 of GSAM DATABASE for GSAMID: ',gsamid
      call skiplines(i0,12)
      goto 234
 202  read(i0,*)
      write(i9,2) 'Error in Line 2 of GSAM DATABASE for GSAMID: ',gsamid
      call skiplines(i0,11)
      goto 234
 203  read(i0,*)
      write(i9,2) 'Error in Line 3 of GSAM DATABASE for GSAMID: ',gsamid
      call skiplines(i0,10)
      goto 234
 204  read(i0,*)
      write(i9,2) 'Error in Line 4 of GSAM DATABASE for GSAMID: ',gsamid
      call skiplines(i0,9)
      goto 234
 205  read(i0,*)
      write(i9,2) 'Error in Line 5 of GSAM DATABASE for GSAMID: ',gsamid
      call skiplines(i0,8)
      goto 234
 206  read(i0,*)
      write(i9,2) 'Error in Line 6 of GSAM DATABASE for GSAMID: ',gsamid
      call skiplines(i0,7)
      goto 234
 207  read(i0,*)
      write(i9,2) 'Error in Line 7 of GSAM DATABASE for GSAMID: ',gsamid
      call skiplines(i0,6)
      goto 234
 208  read(i0,*)
      write(i9,2) 'Error in Line 8 of GSAM DATABASE for GSAMID: ',gsamid
      call skiplines(i0,5)
      goto 234
 209  read(i0,*)
      write(i9,2) 'Error in Line 9 of GSAM DATABASE for GSAMID: ',gsamid
      call skiplines(i0,4)
      goto 234
 210  read(i0,*)
      write(i9,2) 'Error in Line 10 of GSAM DATABASE for GSAMID ',gsamid
      call skiplines(i0,3)
      goto 234
 211  read(i0,*)
      write(i9,2) 'Error in Line 11 of GSAM DATABASE for GSAMID ',gsamid
      call skiplines(i0,2)
      goto 234
 212  read(i0,*)
      write(i9,2) 'Error in Line 12 of GSAM DATABASE for GSAMID ',gsamid
      call skiplines(i0,1)
      goto 234
 213  read(i0,*)
      write(i9,2) 'Error in Line 13 of GSAM DATABASE for GSAMID ',gsamid
      goto 234
 2    format(1x,a,1x,a)
      end
```

SUB-PROGRAM SKIPLINES()

LOCATION: READONE.FOR

MAIN THEME: This routine is invoked in READONE.FOR for skipping necessary number of lines for a reservoir which has errors. After skipping the necessary number of lines, the next record is read in READONE.FOR.

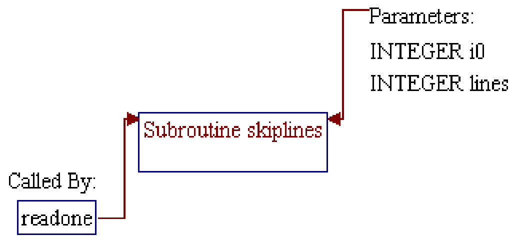
CALLS: None

CALLED BY: READONE() (in file READONE.FOR)
Reads one record from the full database format .GSM file.

READS: [GSAM].GSM
(Discovered reservoir database files)

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variable are declared.**

Note: Name of the sub-program is SKIPLINES() and the parameters passed to this sub-program are as follows:

• *i0*	Unit number for input file [GSAM].GSM.
• *Lines*	Number of lines in input file [GSAM].GSM to be skipped.

```
      SUBROUTINE Skiplines (i0,Lines)
```

Note: Local variable is declared.

```
      Character*79 Dum$
```

Step 2: **The following Do Loop is for number of lines that need to be skipped. The Read statement gets executed *Lines* times before getting out the loop. Once all the lines are read, the control in the main calling program starts fresh from the next GSAMID.**

```
      Do I = 1, Lines
              Read (i0,100) Dum$
      End Do
100   Format (A79)
```

Step 3: **The program control is returned back to the calling routine (sub-program READONE()) and the sub-program SKIPLINES() is ended.**

```
      Return
      End
```

SUB-PROGRAM CNTRL()

LOCATION: MODULE6D.FOR

MAIN THEME: This routine initializes minimum pressures, maximum rates, skins, infill wells ON/OFF, etc.

CALLS: CALCOF() (in file MODULE6B.FOR)
Calculates open flow potentials.

CONVLV () (in file MODULE6C.FOR)
Performs numerical convolution to determine pressure drop caused by previous production.

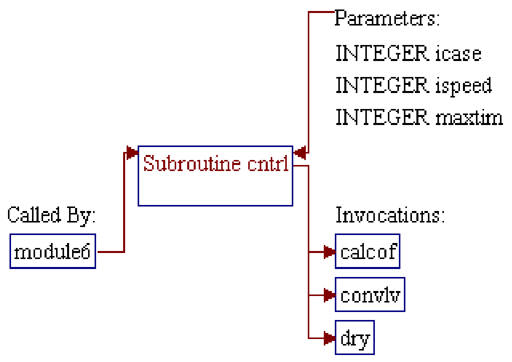
DRY() (in file MODULE6A.FOR)
Controls sub-routine DRY() to calculate gas flow rates for dry coal and dry shale reservoirs.

CALLED BY: MODULE6() (in file MODULE6A.FOR)
Controls the type curve modules in generating type curve data.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included.**

Note: Name of the sub-program is CNTRL() and the parameters passed to this sub-program are as follows:

- *ICase* Case number: 1=primary, 2=automatic refrac, 3=automatic infill one time.
- *MaxTim* Maximum number of time steps
- *ISpeed* Flag for standard formula (*ISpeed=0*, higher accuracy) or simplified formula (*ISpeed=1*, lower accuracy, faster)

```
      SUBROUTINE Cntrl  (ICase,  MaxTim, ISpeed)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'type3.h'
      include 'type5.h'
      include 'type4.h'
      include 'type7.h'
      include 'type8.h'
      include 'type10.h'
```

Step 2: **Minimum wellhead pressure (*Pmin()*), maximum flow rate (*QMax()*), gas production rate (*Qg()*), water production rate (*Qw()*), cumulative gas production (*CumGas()*), change in production rate (*DQ()*), absolute open flow on an annual basis (*CAOF()*), average reservoir pressure (*PreAvg()*), water influxes (*WtrInf(), We(), WePrev(), Wp()*), and shut-in flag (*KShut()*) are initialized.**

Note: *PreMin* is a user specified minimum wellhead pressure and *Pinit()* is initial reservoir pressure.

```
      Do I = 1, 3
              Do K = 1, MaxTim
                      Do J = 1, 3
                              Pmin  (I, J, K) = PreMin
                              QMax  (I, J, K) = 0.
                              Qg    (I, J, K) = 0.
                              Qw    (I, J, K) = 0.
                              CumGas(I, J, K) = 0.
                              DQ    (I, J, K) = 0.
                              CAOF  (I, J, K) = 0.
                      End Do
                      PreAvg(I,K) = 0.
                      WtrInf(I,K) = 0.
                      If (K.eq.1) PreAvg(I,K) = Pinit(I)
```

```
                End Do
                We(I)     = 0.
                WePrev(I) = 0.
                Wp(I)     = 0.
                KShut(I)  = 0
        End Do
```

Step 3: **Maximum rate constraint (*RatMax*) is set.**

Note: The *RatMax* is only set for primary well case (ICase=1). *RatMax* is set to 1 if the specified value is negative.

```
        If (ICase.eq.1) then
                If (RatMax .lt. 0.) RatMax = 1.
```

Note: If the specified *RatMax* is a fraction, the maximum rate is calculated as fraction of initial absolute open flow (*CAOF()* at first time step). For the purpose of calculating *CAOF()* at the first time step, time step number (*ITime*) is set to 1 (first time step), iteration level (*J*) is set to 1 (first iteration), first time step (*Time(1)*) is stored and the value is temporarily set to 1 year.

```
                ITime = 1
                J = 1
                Tim = Time(1)
                Time(1) = 1.
                If (RatMax .le. 1.001) then
```

Note: For dry coal/shale reservoirs (Reservoir Module 6 with *KUnCon()=0* for coal or *2* for shale), the *CAOF()* is calculated by invoking sub-program DRY() with very high maximum rate constraint (*RatMax=1000000* MCFD). In order to keep the value of *RatMax*, its value is first stored to temporary variable *Frac* and will be recalled back after invoking sub-routine DRY(). The module number is obtained from variable *IMod()*.

```
                    K6 = (IMod(1,1)-6) * (IMod(2,1)-6) * (IMod(3,1)-6)
                    If (K6 .eq. 0) then
                        Frac = RatMax
                        RatMax = 1000000.
                        If ((KUnCon(1).eq.0).or.(KUnCon(1).eq.2))
     +                       Call DRY (ITime,ICase,IChg,MaxTim,IOF)
                        RatMax = Frac
```

Note: For other reservoirs, the *CAOF()* at the first time step is calculated by invoking sub-programs CONVLV() and CALCOF().

```
                    Else
                        Call CONVLV (ITime, J, ISpeed)
```

```
                         Call CALCOF (ITime, ICase, IChg, ISpeed)
                     End If
```

Note: The maximum rate constraint (*RatMax*) is then calculated as fraction of initial absolute open flow (*CAOF()* at first time step). Number of wells is calculated by dividing drainage area (*Area()*) with well spacing (*WSpace()*).

```
                     AOF = CAOF(1,1,1)*Area(1)/WSpace(1) +
      +                    CAOF(2,1,1)*Area(2)/WSpace(2) +
      +                    CAOF(3,1,1)*Area(3)/WSpace(3)
                     RatMax = RatMax * AOF
```

Note: For wet coal/shale reservoirs (Reservoir Module 6 with *KUnCon()=1* for coal or *3* for shale), the maximum rate constraint *RatMax* is set to a big number (*RatMax=1000000* MCFD).

```
                     If ((K6.eq.0).and.((KUnCon(1).eq.1).or.
      +                  (KUnCon(1).eq.3))) RatMax = 1000000.
                 End If
```

Note: The first time step (*Time(1)*) is set to its original value.

```
                 Time(1) = Tim
         End If
```

Step 4: **Program control is returned back to the calling routines (sub-program MODULE6()) and the sub-program CNTRL() is ended.**

```
         Return
         End
```

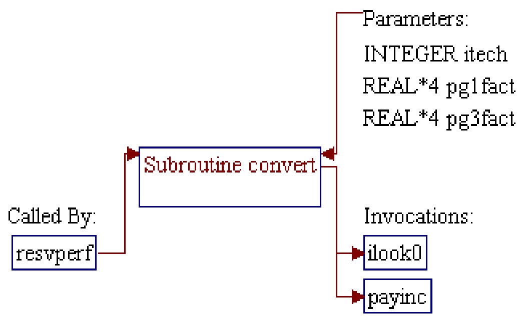
SUB-PROGRAM CONVERT()

LOCATION:	CONVERT.FOR
MAIN THEME:	This routine converts the .GSM data into type curve variable names and distributes them on a pay grade level.

CALLS:	ILOOK0() (in file IOFUNCT.FOR)
	Searches location of an integer number in a set of array.
	
	PAYINC() (in file CONVERT.FOR)
	Sets a factor based on well spacing to adjust the percentage of total pay that is in contact with a well.

CALLED BY:	RESVPERF (in file RESVPERF.FOR) Main program of Reservoir Performance Module.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".H" files are included. Additional common blocks and local variables are declared**

Note: Name of the sub-program is CONVERT() and the parameters passed to this sub-program are as follows:

- *itech* *Technology flag (1=current, 2=advanced)*
- *pg1fact, pg3fact* Factors to reduce or increase drainage area of pay grades 1 and 3. Currently these factors are not implemented (set to be zero).

```
      subroutine convert(itech,pg1fact,pg3fact)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'global.h'
      include 'gsamvar.h'
      include 'field.h'
      include 'welldata.h'
      include 'geology.h'
      include 'type_out.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
      include 'tech.h'
```

Note: Additional common blocks and local variables are declared.

```
      common /block1/ p1cum(3)
      common/skinvalue/ skinfac(3)
      integer i,j
      integer itech,iwlspac
      common/scale/s_gip(3)
      common /num_well/ nwella(3)
      real*4 pg1fact,pg3fact,fac_horz
```

Step 2: **Factors *pg1fact* and *pg3fact* are set to zero.**

```
      pg1fact=0.
      pg3fact=0.
```

Step 3: **Sub-program ILOOK0() is invoked to locate resource type pointer (*irestype*) in reservoir type map array (*res_map()*) that represents Reservoir Module number given in variable *module.* If the location is not found, a default value is assigned (i.e. *qrestype+1*).**

Note: Entries for *res_map()* array and number of entries in the array (*nrestype*) are obtained from input file GEOLOGY.DAT.

```
       irestype=module
       call ilook0(module,res_map,nrestype,irestype)
       if(irestype.eq.0) irestype=qrestype+1
```

Step 4: **Gas gravity (*gasgrv1*), bottom hole temperature (*tem*), impurity concentrations (*cnch2s*, *cncco2*, and *cncn2*) are assigned.**

Note: Minimum value for gas gravity is set to 0.6.

```
       gasgrv1=max(gasgrv,0.60)
       tem=bhtemp
       cnch2s=h2s
       cncco2=co2
       cncn2=n2
```

Step 5: **Ranges for production tubing inside diameter (*diam*), total porosity (*portot*), horizontal permeability (*perhor*), well spacing (*wlspac*), and net pay thickness (*netpay*) are verified. If the value is out of range, default value is used.**

Note: *gsamsr* is GSAM supply region code.

```
       if(diam.le.0.0)
      $  diam=diam_tech(itech,gsamsr)
       if(diam.le.0.0) diam=diam_tech(itech,qreg)
       if (portot.le.0.0) portot=.1
       if (perhor.le.0.0) perhor = 0.01
       if (wlspac.le.0.) wlspac=80.
       if (netpay.le.0.) netpay=10.
```

Step 6: **Aquifer type (*kaqtyp()*), trapped gas saturation behind advancing water influx front (*sgtrap()*), and maximum water rate per well (*qwmax()*) are set.**

Note: *kaqtyp()* can have a value of *0* for finite reservoir and *1* for infinite reservoir and it is obtained from input variable *aqurad*. *sgtrap()* is defaulted to 20%. *qwmax()* is set to negative value (*-0.1*) to indicate that the unit is (BBL/MCF). Here the *qmax()* is defaulted to 0.1 BBL/MCF.

```
         do 987 i =1,3
          kaqtyp(i) = int(aqurad)
          sgtrap(i) = 0.20
          qwmax(i) =  -0.1
  987    continue
```

Step 7: **Value for well spacing (*wlspac1*, *iwlspac*) is assigned. For horizontal well with lateral length of higher than 500 ft, the well spacing is modified accordingly. The modification is performed to all Reservoir Modules except for Module 6 (radial flow in unconventional gas reservoirs).**

```
             wlspac1 = wlspac
             iwlspac = wlspac
             if(module.ne.6.and.jlen_tech(2,gsamsr).ge.500)then
              wlspac1 = wlspac + 2*jlen_tech(1,gsamsr)*
      @                 sqrt(wlspac/(43560*3.14159))
             endif
```

Step 8: **Reservoir properties in each pay grade are assigned.**

Note: Reservoir properties assigned in this section are:

- *pinit()* Initial reservoir pressure (psia)
- *perm()* Horizontal permeability (md). The value is set equal to horizontal permeability from [GSAM].GSM file (*perhor*), multiplied by pay grade permeability factor (*perm_fac()*) specified in input file GEOLOGY.DAT.
- *permv()* Vertical permeability (md). The value is set equal to vertical permeability from [GSAM].GSM file (*perver*), multiplied by pay grade permeability factor (*perm_fac()*) specified in input file GEOLOGY.DAT. The value of horizontal permeability is used if the vertical permeability is not specified (negative).
- *poros()* Porosity (%). The value is set equal to total porosity from [GSAM].GSM file (*portot*), multiplied by pay grade porosity factor (*por_fac()*) specified in input file GEOLOGY.DAT.
- *swi()* Initial water saturation (fraction). The value is set equal to water saturation from [GSAM].GSM file (*watsat*), multiplied by pay grade water saturation factor (*h2osat_fac()*) specified in input file GEOLOGY.DAT.

- *thick()* Net pay thickness (ft). The value is set equal to net pay thickness from [GSAM].GSM file (*netpay*), multiplied by pay grade net pay thickness factor (*netpay_fac()*) specified in input file GEOLOGY.DAT.
- *salin()* Water salinity (ppm)
- *permma()* Matrix permeability (md). The value is set equal to matrix permeability from [GSAM].GSM file (*permtx*), multiplied by pay grade permeability factor (*perm_fac()*) specified in input file GEOLOGY.DAT.
- *porma()* Matrix porosity (%). The value is set equal to matrix porosity from [GSAM].GSM file (*pormtx*), multiplied by pay grade porosity factor (*por_fac()*) specified in input file GEOLOGY.DAT.
- *area()* Total area within pay grade (acres). The value is set equal to area from [GSAM].GSM file (*acprod*), multiplied by pay grade area factor (*area_fac()*) specified in input file GEOLOGY.DAT.
- *frcspc()* Natural fracture spacing (ft)
- *depth1()* Reservoir depth (ft)
- *wspace()* Initial well spacing (acres)
- *pl()* Langmuir pressure (psia)
- *tdes()* Sorption time constant (days). Minimum value for sorption time constant is set to 50 days.
- *gascon1()* Initial coal/shale gas content (MSCF/ton).
- *rhoma()* Matrix density (gr/cc). The value is initialized to zero.
- *kuncon()* Type of unconventional reservoir: 0=dry coal, 1=wet coal, 2=dry shale, 3=wet shale.
- *iloc()* Coal location: 0=Eastern U.S., 1=Western U.S.

```
      do ipay=1,3
       pinit(ipay)= presin
       perm(ipay)=perm_fac(ipay,irestype)*perhor
       permv(ipay)=perm_fac(ipay,irestype)*pervrt
       if (permv(ipay).le.0.0) permv(ipay) = perm(ipay)
       poros(ipay)=por_fac(ipay,irestype)*portot
       swi(ipay)=h2osat_fac(ipay,irestype)*watsat
       thick(ipay)=netpay_fac(ipay,irestype)*netpay
       salin(ipay)=chlcon
       permma(ipay)=perm_fac(ipay,irestype)*permtx
       porma(ipay)=por_fac(ipay,irestype)*pormtx
       area(ipay)=area_fac(ipay,irestype)*acprod
       frcspc(ipay)=fracsp
       depth1(ipay)=depth
       wspace(ipay)=wlspac1
       pl(ipay)=langpr
       tdes(ipay)=srptim
c       addition by vikas on 11/21/94 at 9:00 AM
       if (tdes(ipay).le.0.0) tdes(ipay) = 50.0
       gascon1(ipay) = gascon
       rhoma(ipay)=0.0
       kuncon(ipay) = iunctype
       iloc(ipay) = iuncloc
      enddo
```

Step 10: Original gas in place in each pay grade is calculated and stored in variable *s_gip()*.

```
      do ipay=1,3
       s_gip(ipay)=area(ipay)*thick(ipay)*poros(ipay)*(1-swi(ipay))
      enddo
```

Step 11: **Number of wells in pay grades 1 and 3 (*nwella(1)* and *nwella(3)*) are calculated based on well spacing constraint.**

```
      nwella(1)= int(area(1)/wspace(1))
      nwella(3)= int(area(3)/wspace(3))
```

Step 12: **Residual gas in place from pay grades 1 and 3 is added to pay grade 2.**

```
      s_gip(2)=s_gip(2)+
     &   s_gip(1)-(nwella(1)*wspace(1))*thick(1)*poros(1)*(1-swi(1))+
     &   s_gip(3)-(nwella(3)*wspace(3))*thick(3)*poros(3)*(1-swi(3))
```

Step 13: **Residual area from pay grades 1 and 3 is added to pay grade 2. Areas for pay grades 1 and 3 are recalculated based on number of wells in Step 11 (integer number) and well spacing. In this way, the area will be consistent with number of wells and well spacing.**

```
      area(2)=area(2)+
     &   area(1)-wspace(1)*nwella(1)+
     &   area(3)-wspace(3)*nwella(3)
      area(1)=wspace(1)*nwella(1)
      area(3)=wspace(3)*nwella(3)
```

Step 14: **Number of wells in pay grade 2 (*nwella(2)*) is calculated based on well spacing constraint.**

```
      nwella(2)=nint(area(2)/wspace(2))
```

Step 15: **Check value of well spacing in pay grade 2. If it is higher than the drainage area, the well spacing is set equal to drainage area and number of well is set to one.**

```
        if (area(2).le.wspace(2)) then
           wspace(2) = area(2)
           nwella(2) = 1
         endif
```

Step 16: **Area of pay grade 2 is recalculated based on number of wells and well spacing similar to the calculation in Step 13. If value of drainage area is negative (if well spacing and/or number of wells in pay grade 2 are not specified), the value is defaulted to 320 acres.**

```
       area(2)=wspace(2)*nwella(2)
       if (area(2).le.0.) area(2) = 320.0
```

Step 17: **Maximum value for initial water saturation in pay grade 2 (*swi(2)*) is set to 99%.**

```
       if (swi(2).eq.1.0) swi(2) = 0.99
```

Step 18: **Except for Reservoir Module 6 (radial flow in unconventional gas reservoirs), net pay thickness of pay grade 2 (*thick(2)*) is calculated based on the adjusted gas in place, drainage area, porosity, and initial water saturation.**

```
       IF(MODULE.LT.6)THEN
         thick(2)=s_gip(2)/(area(2)*poros(2)*(1-swi(2)))
       ENDIF
```

Step 19: **For fractured vertical wells, fracture half length *halfln()*) and fracture conductivity (*cond()*) are assigned with values obtained from input file TECH.DAT (*fracxf_tech()* and *fraccn_tech()*, respectively).**

Note: These assignments are performed for all Reservoir Modules except for Modules 1 and 5 (radial flows in conventional and water drive gas reservoirs, respectively). Well type is set to vertical (*jtyp()=0*). For Appalachia (GSAM supply region equals to 1, *gsamsr=1*), the fracture half length is reduced by half.

```
       do 101 i=1,3
        do 101 j=1,3
         jtyp(i,j)=0
         if(module.ne.1.or.module.ne.5)then
           halfln(i,j) =fracxf_tech(itech,module)
           if(gsamsr.eq.1)halfln(i,j)=halfln(i,j)*.5
           cond(i,j) = fraccn_tech(itech,module)
         endif
  101   continue
```

Step 20: **Skin factor for each pay grade (*skinfac()*) is set equal to value specified in input file TECH.DAT (*fracsk_tech()*). Minimum value for skin factor is set to –5.**

```
        do 103 i=1,3
         skinfac(i) = fracsk_tech(itech,module)
         if(skinfac(i).lt.-5.)skinfac(i)=-5
 103    continue
```

Step 21: **Value for horizontal well properties are assigned. The assignments are skipped for Reservoir Module 6 (radial flow in unconventional gas reservoirs).**

Note: Variable *jtyp_tech()* is a well type flag specified in file TECH.DAT. For horizontal wells, *jtyp_tech()* is greater or equal to one (*jtyp_tech()*>=1). Since horizontal well is modeled as infinite conductivity fracture, all modules for radial flows (Reservoir Modules 1, 2, and 5) are set to linear flow in Module 2 (linear flow in conventional gas reservoirs). Well type is set to horizontal (*JTyp()=1*), fracture half length is set equal to lateral length of the well, and the fracture conductivity is set to 1E6 md-ft (infinite conductivity fracture).

```
        if (module.eq.6) goto 100
        if(jtyp_tech(itech,gsamsr).ge.1)then
        if (module.ne.2.or.module.ne.4) module=2
        Do I=1, 3
                Do J=1, 3
                        JTyp(I,J)=1
                                HorLen(I,J) = Jlen_tech(itech,gsamsr)
                                halfln(i,j) = horlen(i,j)
                                Cond(I,J)   = 1.e6
                End Do
        End Do
        endif
```

Note: Skin factor for horizontal well is calculated.

```
       if (jtyp_tech(itech,gsamsr).ge.1) then
        do i = 1,3
         fac_horz = (perm(i)/permv(i))**0.50
    @                *thick(i)/Jlen_tech(itech,gsamsr)
         skinfac(i) = fracsk_tech(itech,module)*fac_horz
         if(skinfac(i).lt.-5.)skinfac(i)=-5
        enddo
       endif
```

Step 22: **Maximum rate from field (*ratmax*) is set to a fraction of initial open flow potential (*0<ratmax<1*). This is done by first setting *ratmax* with proration factor specified in input file TECH.DAT**

(*prorat_tech()*). If pay enhancement proration based on BEG study (*proration()*) is specified (also in input file TECH.DAT), the *ratmax* is then set to this proration. In the case of negative *ratmax* (*proration()* is not specified), the *ratmax* is set to a default value specified in *prorat_tech(qstate)*.

```
 100   ratmax = prorat_tech(itech,gsamsr)
       do ist=1,ntech_st
         if(state.eq.tech_st(ist))ratmax=proration(itech,ist)
       enddo
       if(ratmax.le.0.0)ratmax=prorat_tech(itech,qstate)
```

Step 23: **Skin factors for primary (*skin(#,1,1)*), first infill (*skin(#,2,1)*), second infill (*skin(#,3,1)*), and refract (*skin(#,1,2)*) wells are assigned.**

Note: Data for base skin factor is obtained from *skinfac()* and it is assigned to primary well skin (*skin(#,1,1)*). For infill wells (first and second infills), the skin factor is reduced by -1. For *refrac* well, the skin factor is reduced by -3.

```
       do 104 j=1,3
         skin(j,2,1)=skinfac(j)-1
         skin(j,3,1)=skinfac(j)-1
         skin(j,1,1)=skinfac(j)
         skin(j,1,2)=skinfac(j)-3
  104  continue
```

Step 24: **Time step sizes (*time()*) are set to 1 year.**

```
       do 106 n=1,nyr
        time(n)=n*1.0
  106  continue
```

Step 25: **Wellbore radius from input file TECH.DAT (*wrad_tech()*) is used if data of wellbore radius from [GSAM].GSM (*welrad*) larger than *wrad_tech()*. Minimum value for wellbore radius is set to 0.354 ft.**

```
       if(wrad_tech(itech,module).gt.welrad)welrad=
     &   wrad_tech(itech,module)
       if(welrad.lt.0.1)welrad=.354
```

Step 26: **Type curve variables for Reservoir Module flag (*IMod()*), wellbore radius (*rw()*), and factor to adjust the percentage of total pay in contact with the well (*rifact()*) are assigned.**

Note: Values from *module* for Reservoir Module flag and *welrad* for wellbore radius are set to type curve variables *IMod()* and *rw(),* respectively. Sub-program PAYINC() is invoked to set the value of *rifact()* based on the size of well spacing.

```
      do 107 j=1,3
       do 107 i=1,3
        IMod(i,j) = module
        rw(i,j) = welrad
        if (module.eq.6.and.gsamsr.lt.22) then
        rifact(i,j) = 1
        else
        rifact(i,j) = payinc(i,j,wlspac,plycod,itech,module,gsamid)
        endif
 107  continue
```

Step 27: **Minimum wellhead pressure (*premin*) is set to the smallest value between minimum system pressure from file TECH.DAT (*psys_tech()*) and 20% of initial pressure from file [GSAM].GSM (0.2*presin*). If *psys_tech()* value is not given in TECH.DAT (negative pressure), default value specified in *psys_tech(itech,qreg)* is utilized.**

```
      psysc = psys_tech(itech,gsamsr)
      if(psysc.le.0.0)psysc=psys_tech(itech,qreg)
      premin = min(0.2*presin,psysc)
```

Step 28: **Average depth (*avdep*) is set to the depth to the center of the reservoir. This is done by adding half of the thickness (0.5*netpay*) to the reservoir top depth (*depth*).**

```
      avdep=(depth + 0.50*netpay)
```

Step 29: **If database value for water depth (*h2odep*) is missing (negative value), water depth for West Florida (region 14), Norphlet (region 15), Gulf of Mexico East (region 16), and Gulf of Mexico West (region 17) is set to 250 feet.**

Note: GSAM region is obtained from the first 2 digits of the 11-digit GSAMID (*gsamid*).

```
       if(h2odep.le.0.0) then
       if(gsamid(1:2).eq.'14' .or.gsamid(1:2).eq.'15'
     &    .or.gsamid(1:2).eq.'16'.or.gsamid(1:2).eq.'17') h2odep=250
       endif
```

Step 30: **Minimum value for drainage area (*area()*) is set to 0.0001 acres.**

```
      if(area(1).le.0.0) area(1)=0.0001
      if(area(2).le.0.0) area(2)=0.0001
      if(area(3).le.0.0) area(3)=0.0001
```

Step 31: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program CONVERT() is ended.**

```
      return
      end
```

SUB-PROGRAM GET_TYPE()

LOCATION: MODULE6D.FOR

MAIN THEME: This routine assigns number of wells, original gas in place, gas production, and well sandface pressures to type curve variables.

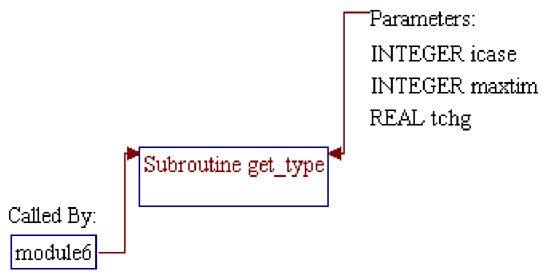
CALLS: None

CALLED BY: MODULE6() (in file MODULE6A.FOR)
Controls the type curve modules in generating type curve data.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variables are declared. Header ".h" files are included and local variables and additional common blocks are defined.**

Note: Name of the sub-program is GET_TYPE() and the parameters passed to this sub-program are as follows:

- *MaxTim* Maximum number of time steps
- *ICase* Case number: 1=primary, 2=automatic refrac, 3=automatic infill one time.
- *TChg* Time at which automatic change in development type occurs (automatic infill or refrac)

```
      subroutine get_type(maxtim,icase,tchg)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'welldata.h'
      include 'type_out.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
```

Note: Local variables and additional common block are declared.

```
      common / stchg/ iwin_yr
      integer icase
      real*4 wells(3,3)
```

Step 2: **Number of wells in each pay grade (Wells()) is set/calculated.**

Note: Number of wells is calculated by dividing drainage area (*Area()*) with well spacing (*WSpace()*). For primary well cases (*ICase=1* or *2*), number of wells is set to zero. For infill wells (*ICase=3*) number of wells is twice of the primary wells.

```
        Do I=1,3
                Wells(I,1) = Area(I) / WSpace(I)
                If ((ICase .eq. 1) .or. (ICase .eq. 2)) then
```

```
                        Wells(I,2) = 0.
                        Wells(I,3) = 0.
                Else
                        Wells(I,2) = Wells(I,1)
                        Wells(I,3) = 0.
                        If (ICase .eq. 4) Wells(I,3) = Wells(I,1) * 2.
                End If
        End Do
```

Step 3: **Number of wells, original gas in place, gas production, and well sandface pressure are calculated and stored in type curve variables.**

Note: *type_well(), type_ogip(), type_gas(),* and *type_bhp()* are type curve variables for number of wells, original gas in place, gas production, and sandface pressure, respectively.

```
         do i=1,3
          do j=1,3
           type_ogip(icase,i)=
     &        ogip1(I) * Wells(I,1) / 1.e6
           do k=1,maxtim
             type_gas(icase,i,k)=type_gas(icase,i,k)+
     &          Qg(I,J,K) * Wells(I,J)*365/1.e6
             if(k.eq.1) type_well(icase,i)=
     &         type_well(icase,i)+wells(i,j)
           enddo
          enddo
         enddo
         Do K = 1, MaxTim
          do i=1,3
           type_pbhp(icase,i,k)=prbh(i,1,k)
           type_pwhp(icase,i,k)=prwh(i,1,k)
           type_ibhp(icase,i,k)=prbh(i,2,k)
          enddo
         End Do
```

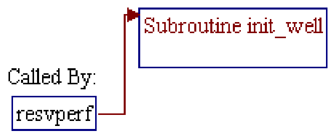
Step 4: **Program control is returned back to the calling routines (sub-program MODULE6()) and the sub-program PRESUR() is ended.**

```
         Return
         End
```

SUB-PROGRAM INIT_WELL

LOCATION:	GSAM_B.FOR
MAIN THEME:	This routine initializes type curve and economic variables.
CALLS:	None
CALLED BY:	RESVPERF (in file RESVPERF.FOR) Main program of Reservoir Performance Module.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name of the sub-program is declared. Header ".h" files are included. Local variables are declared.**

Note: Name of the sub-program is INIT_WELL.

```
      subroutine init_well
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'welldata.h'
      include 'type_out.h'
      include 'cost.h'
```

Note: Local variables are declared.

```
      integer iyr,icase,ipay
```

Step 2: **Type curve and economic variables are initialized to zeros.**

Note: Name and descriptions of the type curve variables are as follows:

• *type_gas()*	Gas production (BCF)
• *type_pbhp()*	Bottomhole pressures of primay wells (psia)
• *type_pwhp()*	Wellhead pressures of primay wells (psia)
• *type_ibhp()*	Bottomhole pressures of inifill wells (psia)
• *type_well()*	Number of wells could be drilled
• *type_ogip()*	Original gas in place (BCF)

Name and descriptions of the economic variables are as follows:

• *masp()*	Minimum acceptable supply price ($/MCF)
• *tot_cap_2()*	Total capital at gas price of $2/MCF ($MM)
• *udatcf_2()*	Undiscounted cash flow at gas price of $2/MCF ($MM)
• *udbtcf_2()*	Undiscounted before tax cash flow at gas price of $2/MCF ($MM)
• *dbtcf_2()*	Discounted before tax cash flow at gas price of $2/MCF ($MM)
• *datcf_2()*	Discounted after tax cash flow at gas price of $2/MCF ($MM)
• *tot_cap_5()*	Total capital at gas price of $5/MCF ($MM)
• *udatcf_5()*	Undiscounted cash flow at gas price of $5/MCF ($MM)
• *udbtcf_5()*	Undiscounted before tax cash flow at gas price of $5/MCF ($MM)
• *dbtcf_5()*	Discounted before tax cash flow at gas price of $5/MCF ($MM)
• *datcf_5()*	Discounted after tax cash flow at gas price of $5/MCF ($MM)

```
      do icase=1,3
       do ipay=1,3
        do iyr=1,qyr
          type_gas(icase,ipay,iyr)=0.0
          type_pbhp(icase,ipay,iyr)=0.0
          type_pwhp(icase,ipay,iyr)=0.0
          type_ibhp(icase,ipay,iyr)=0.0
        enddo
        type_well(icase,ipay)=0.0
        masp(icase,ipay)=0.0
        type_ogip(icase,ipay)=0.0
        tot_cap_2(icase,ipay)=0.0
        udatcf_2(icase,ipay)=0.0
        udbtcf_2(icase,ipay)=0.0
        dbtcf_2(icase,ipay)=0.0
        datcf_2(icase,ipay)=0.0
        tot_cap_5(icase,ipay)=0.0
        udatcf_5(icase,ipay)=0.0
        udbtcf_5(icase,ipay)=0.0
        dbtcf_5(icase,ipay)=0.0
        datcf_5(icase,ipay)=0.0
       enddo
      enddo
```

Step 3: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program INIT_WELL() is ended.**

```
      return
      end
```

SUB-PROGRAM INITCASH

LOCATION: INITIAL.FOR

MAIN THEME: This routine initializes cash flow variables as declared in header file CASHFLOW.H.

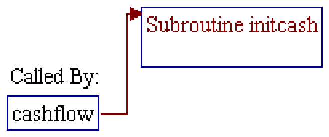
CALLS: None

CALLED BY: CASHFLOW() (in file CASHFLOW.FOR)
Performs a discounted cash flow analysis (i.e. performs a pro-forma cash flow analysis for every reservoir processed).

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name of the sub-program is declared. Header ".h" files are included. Local variable is declared.**

Note: Name of the sub-program is INITCASH.

```
      subroutine initcash
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'cashflow.h'
```

Note: Local variable is declared.

```
      integer iyr
```

Step 2: **Cash flow variables are initialized to zeros.**

Note: Name and descriptions of the cash flow variables are as follows:

- *adjgross()* Adjusted gross revenues
- *netsales()* Net sales
- *toc()* Total operating cost
- *ga_exp()* G&A on expensed items
- *ga_cap()* G&A on capitalized items
- *ii()* Intangible investment
- *intcap()* Portion of intangible to capitalize
- *ti()* Total investments
- *tci()* Depreciable/capitalized investments
- *tciadj()* Depreciable/capitalized investments adjusted for federal tax credits
- *cap_base()* Depreciable/capitalized base
- *depr()* Depreciation
- *dggla()* Depletion base
- *dep_crd()* Adjustments for tax credits (depletable G&G/lease costs)
- *eggla()* Expensed G&G/lease costs
- *deplet()* Depletion allowance
- *apd()* Allowable percent depletion
- *nilb()* Net income limitation base
- *eortca()* EOR tax credit addback
- *idca()* Intangible drilling cost addback

- *oia()* Other intangible addbacks
- *iea()* Intangible environmental addback
- *eoca()* Environmental operating cost addback
- *intadd()* Total intangible addback
- *ggla()* G&G/lease addback
- *nibta()* Net income before tax addback
- *nibt()* Net income before taxes
- *sttax()* State income taxes
- *fti()* Federal taxable income
- *fedtax()* Federal income taxes
- *amti()* Alternative minimum taxable income
- *acpamt()* Available credits for past alternative minimum taxable income
- *amint()* Alternative minimum taxes
- *ace()* ACE
- *uamti()* Unadjusted available minimum taxable income
- *eidca()* Excess intangible drilling cost addback
- *nifoag()* Net income from oil and gas
- *dpidcs()* Deduction portion of IDC costs
- *idcpamt()* IDC preference for alternative minimum taxable income
- *aceadj()* ACE adjustment
- *fedtaxc()* Federal tax credits
- *niat()* Net income after taxes
- *aatcf()* Annual after tax cash flow
- *datcf()* Discounted after tax cash flow
- *catcf()* Cumulative after tax cash flow
- *sevtax()* Severance taxes
- *tfit()* Tentative federal income taxes
- *sfit()* Selected federal income taxes
- *ucpamt()* Useable credits for past AMT
- *bamtp()* Balance of AMT paid
- *lastyr* Last year of operation
- *intang_ewc()* Intangible exploratory cost
- *intang_dwc()* Intangible development cost
- *tang_ewc()* Tangible exploratory cost
- *tang_dwc()* Tangible development cost

```
       do iyr=1,qyr
        adjgross(iyr)=0.0
        netsales(iyr)=0.0
        toc(iyr)=0.0
        ga_exp(iyr)=0.0
        ga_cap(iyr)=0.0
        ii(iyr)=0.0
        intcap(iyr)=0.0
        ti(iyr)=0.0
        tci(iyr)=0.0
        tciadj(iyr)=0.0
        cap_base(iyr)=0.0
```

```
         depr(iyr)=0.0
         dggla(iyr)=0.0
         dep_crd(iyr)=0.0
         eggla(iyr)=0.0
         deplet(iyr)=0.0
         apd(iyr)=0.0
         nilb(iyr)=0.0
         eortca(iyr)=0.0
         idca(iyr)=0.0
         oia(iyr)=0.0
         iea(iyr)=0.0
         eoca(iyr)=0.0
         intadd(iyr)=0.0
         ggla(iyr)=0.0
         nibta(iyr)=0.0
         nibt(iyr)=0.0
         sttax(iyr)=0.0
         fti(iyr)=0.0
         fedtax(iyr)=0.0
         amti(iyr)=0.0
         acpamt(iyr)=0.0
         amint(iyr)=0.0
         ace(iyr)=0.0
         uamti(iyr)=0.0
         eidca(iyr)=0.0
         nifoag(iyr)=0.0
         dpidcs(iyr)=0.0
         idcpamt(iyr)=0.0
         aceadj(iyr)=0.0
         fedtaxc(iyr)=0.0
         niat(iyr)=0.0
         aatcf(iyr)=0.0
         datcf(iyr)=0.0
         catcf(iyr)=0.0
         sevtax(iyr)=0.0
         tfit(iyr)=0.0
         sfit(iyr)=0.0
         ucpamt(iyr)=0.0
         bamtp(iyr)=0.0
         lastyr=1
         intang_ewc(iyr)=0.0
         intang_dwc(iyr)=0.0
         tang_ewc(iyr)=0.0
         tang_dwc(iyr)=0.0
        enddo
```

Step 3: **The program control is returned back to the calling routine (sub-program CASHFLOW()) and the sub-program INITCASH() is ended.**

```
        return
        end
```

SUB-PROGRAM INITCOST

LOCATION: INITIAL.FOR

MAIN THEME: This routine initializes costing variables as declared in header file COSTING.H.

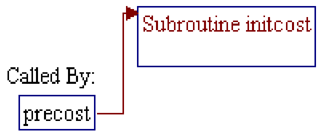
CALLS: None

CALLED BY: PRECOST() (in file PRECOST.FOR)
Utilizes the unit cost data to create the cost streams to be fed to the cash flow routine CASHFLOW().

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name of the sub-program is declared. Header ".h" files are included. Local variable is declared.**

Note: Name of the sub-program is INITCOST.

```
      subroutine initcost
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'costing.h'
```

Note: Local variable is declared.

```
      integer iyr
```

Step 2: **Costing variables are initialized to zeros.**

Note: Name and descriptions of the costing variables are as follows:

- *icap()* Intangible capital (MM$)
- *eicap()* Environmental intangible capital costs (MM$)
- *etcap()* Environmental tangible capital costs (MM$)
- *eoam()* Environmental operation and maintenance costs (MM$)
- *oam()* Operation and maintenance costs (MM$)
- *inj()* Injectant costs (MM$)
- *gravpen()* Gravity penalty (MM$)
- *transcst()* Transportation costs (MM$)
- *gg()* G&G costs (MM$)
- *la()* Lease acquisition costs (MM$)
- *dwc()* Development well costs (MM$)
- *ewc()* Exploratory well costs (MM$)
- *otc()* Other tangible capital (MM$)
- *stim()* Stimulation costs (MM$)
- *comp()* Compressor costs (MM$)
- *recomp()* Recompletion costs (MM$)

```
      do iyr=1,qyr
       icap(iyr)=0.0
       eicap(iyr)=0.0
       etcap(iyr)=0.0
```

```
         eoam(iyr)=0.0
         oam(iyr)=0.0
         inj(iyr)=0.0
         gravpen(iyr)=0.0
         transcst(iyr)=0.0
         gg(iyr)=0.0
         la(iyr)=0.0
         dwc(iyr)=0.0
         ewc(iyr)=0.0
         otc(iyr)=0.0
         stim(iyr)=0.0
         comp(iyr)=0.0
         recomp(iyr)=0.0
        enddo
```

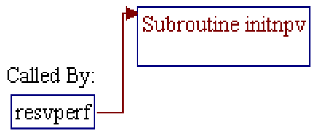
Step 3: **The program control is returned back to the calling routine (sub-program PRECOST()) and the sub-program INITCOST() is ended.**

```
        return
        end
```

SUB-PROGRAM INITNPV

LOCATION:	INITIAL.FOR
MAIN THEME:	This routine initializes net present value (NPV) variables as declared in header file NPV.H.
CALLS:	None
CALLED BY:	RESVPERF (in file RESVPERF.FOR) Main program of Reservoir Performance Module.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name of the sub-program is declared. Header ".h" files are included. Local variable is declared.**

Note: Name of the sub-program is INITNPV.

```
      subroutine initnpv
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'npv.h'
```

Note: Local variable is declared.

```
      integer inpv
```

Step 2: **NPV variables are initialized to zeros.**

Note: Name and descriptions of the NPV variables are as follows:

- *npv()* Net present value (NPV)
- *o_prd_npv()* Oil production NPV
- *q_prd_npv()* Gas production NPV
- *gross_npv()* Gross sales NPV
- *toc_npv()* Total operating cost NPV
- *int_npv()* Intangible investment NPV
- *tan_npv()* Tangible investment NPV
- *dwc_npv()* Development cost NPV
- *ewc_npv()* Exploratory cost NPV
- *tax_npv()* Tax NPV
- *depggla_npv()* Depletable G&G/lease acquisition NPV
- *expggla_npv()* Expensed G&G/lease acquisition NPV
- *credit_npv()* Credits NPV
- *tot_inv()* Total investments NPV
- *totalcst()* Total cost of each scenario

```
      do inpv=1,qnpv
       npv(inpv)=0.0
       o_prd_npv(inpv)=0.0
       g_prd_npv(inpv)=0.0
       gross_npv(inpv)=0.0
       toc_npv(inpv)=0.0
       int_npv(inpv)=0.0
```

```
         tan_npv(inpv)=0.0
         dwc_npv(inpv)=0.0
         ewc_npv(inpv)=0.0
         tax_npv(inpv)=0.0
         depggla_npv(inpv)=0.0
         expggla_npv(inpv)=0.0
         credit_npv(inpv)=0.0
         tot_inv(inpv)=0.0
         totalcst(inpv)=0.0
        enddo
```

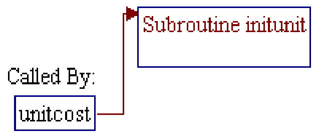
Step 3: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program INITNPV() is ended.**

```
        return
        end
```

SUB-PROGRAM INITUNIT

LOCATION:	INITIAL.FOR
MAIN THEME:	This routine initializes cash flow variables as declared in header file UNITCOST.H.
CALLS:	None
CALLED BY:	UNITCOST() (in file UNITCOST.FOR) Calculates unit costs in $/MCF, $/Well or $/BBL.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name of the sub-program is declared. Header ".h" files are included. Local variable is declared.**

Note: Name of the sub-program is INITUNIT.

```
      subroutine initunit
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'unitcost.h'
```

Note: Local variable is declared.

```
      integer iyr
```

Step 2: **Cash flow variables are initialized to zeros.**

Note: Name and descriptions of the cash flow variables are as follows:

- *ewc_w* EWC unit cost (MM$/well)
- *dwc_w* DWC unit cost (MM$/well)
- *stim_w* Stimulation cost (MM$/well)
- *fac_w* Facilities cost (MM$/well)
- *env_cap_w* Environmental capital cost (MM$/well)
- *fxoam_w* Fixed O&M (MM$/well)
- *voam_g* Surface O&M - gas ($/MCF)
- *h2ooam_w* Surface O&M - water ($/BBL)
- *envni* Intangible environmental new well cost (MM$/well)
- *envnt* Tangible environmental new well cost (MM$/well)
- *envei* Intangible environmental well cost (MM$/well)
- *envet* Tangible environmental well cost (MM$/well)
- *env_oam_g* Environmental O&M - gas ($/MCF)
- *env_oam_w* Environmental O&M - water ($/BBL)
- *env_oam_l* Environmental O&M – well ($/well/year)
- *env_oam_n* Environmental O&M – new well ($/well/year)
- *lbc_frac* Fraction of lease bonus
- *tang_m()* Tangible multiplier
- *intang_m()* Intangible multiplier
- *oam_m()* O&M multiplier

```
       ewc_w=0.0
       dwc_w=0.0
       stim_w=0.0
       fac_w=0.0
       env_cap_w=0.0
       fxoam_w=0.0
       voam_g=0.0
       h2ooam_w=0.0
       envni=0.0
       envnt=0.0
       envei=0.0
       envet=0.0
       env_oam_g=0.0
       env_oam_w=0.0
       env_oam_l=0.0
       env_oam_n=0.0
       lbc_frac=0.0
       do iyr=1,qyr
        tang_m(iyr)=0.0
        intang_m(iyr)=0.0
        oam_m(iyr)=0.0
       enddo
```

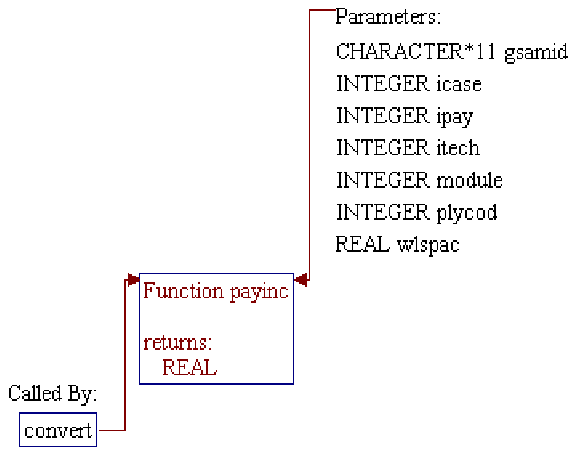
Step 3: **The program control is returned back to the calling routine (sub-program UNITCOST()) and the sub-program INITUNIT() is ended.**

```
       return
       end
```

SUB-PROGRAM PAYINC()

LOCATION:	CONVERT.FOR
MAIN THEME:	This routine sets a fraction of total pay that is in contact with a well, based on well spacing.
CALLS:	None
CALLED BY:	CONVERT() (in file CONVERT.FOR) Converts the .GSM data into type curve variable names and distributes them on a pay grade level.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".H" files are included. Local variables are declared**

Note: Name of the sub-program is PAYINC() and the parameters passed to this sub-program are as follows:

Input Parameters:

- *icase* Development case flag (1=primary, 2=first infill, 3=second infill)
- *ipay* Pay grade number (not currently used)
- *wlspac* Well spacing (acres)
- *plycod* Play code (not currently used)
- *itech* Technology flag (1=current, 2=advanced)
- *module* Reservoir Module number
- *gsamid* 11-digit GSAMID

Output Parameter:

- *payinc* fraction of total pay that is in contact with a well

```
      function payinc(icase,ipay,wlspac,plycod,itech,module,gsamid)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'tech.h'
```

Note: Local variables are declared.

```
      integer plycod,ipay
      character*11 gsamid
```

Step 2: **Factor *payinc* is first defaulted to 0.2.**

Note: Under normal condition and well spacing of 320 acres and larger, the percentage of total pay that is in contact with the well is assumed to be 80% (*payinc=0.8*). Here *payinc* is set to 0.2 as the fraction of total pay that is NOT in contact with the well. Later in the program, the value is adjusted to *1-payinc.*

```
       payinc = 0.20
```

Step 3: **Factor *payinc* is assigned based on region (*gsamid(1:2)*), production status (*gsamid(3:3)*), and well spacing (*wlspac*).**

Note: For producing reservoirs in Appalachia, indicated by *gsamid(1:2)='01'* (code for Appalachia) and *gsamid(3:3)='3'* (code for producing reservoirs), value for factor *payinc* is set to:

- *0.03* 0 <= *wlspac* < 160 acres
- *0.08* 160 <= *wlspac* < 320 acres
- *0.12* 320 <= *wlspac* < 640 acres

For other reservoirs, value for factor *payinc* is set to:

- *0.03* 0 <= *wlspac* < 80 acres
- *0.08* 80 <= *wlspac* < 160 acres
- *0.12* 160 <= *wlspac* < 320 acres

```
if (gsamid(1:2).eq.'01'.and.gsamid(3:3).eq.'3') then
  if (wlspac.lt.640.0) payinc = 0.12
  if (wlspac.lt.320.0) payinc = 0.08
  if (wlspac.lt.160.0) payinc = 0.03
else
  if (wlspac.lt.320.0) payinc = 0.12
  if (wlspac.lt.160.0) payinc = 0.08
  if (wlspac.lt.80.0) payinc = 0.03
endif
```

Step 4: **For second infill case (*icase=3*), factor *payinc* is further reduced by 50%.**

```
if (icase.eq.3) payinc = payinc*0.50
```

Step 5: **Pay continuity enhancement factor (*payenh*) is set to the value from input file TECH.DAT (*pay_tech()*). If the *pay_tech()* is not specified in the file, the default value specified in *pay_tech(itech,qreg)* is utilized.**

```
payenh=pay_tech(itech,module)
if(payenh.le.0.0)payenh=pay_tech(itech,qreg)
```

Step 5: **Value for *payinc* assigned in Steps 2 through 5 is actually the fraction of total pay that is NOT contacted with the well. Therefore, fraction of total pay that is contacted with the well will be *1–payinc*.**

Note: The following code calculates ratio of the previously assigned *payinc* with the pay continuity enhancement factor (*payenh*) and stores the ratio to *payinc*. The true value for *payinc* is then set equal to *1-payinc*.

```
payinc = payinc/payenh
payinc = 1.0 – payinc
```

Step 6: **Program control is returned back to the calling routines (sub-program CONVERT()) and the sub-program PAYINC() is ended.**

```
Return
End
```

SUB-PROGRAM RDUNCN

LOCATION: MODULE6B.FOR

MAIN THEME: This routine assigns sorption properties for unconventional reservoir. The properties are:

• *PL()*	Langmuir pressure (psia)
• *VL()*	Langmuir volume (SCF/ton)
• *TDes()*	Sorption time constant (days)
• *RhoMa*	Matrix density (ton/acre.ft)???

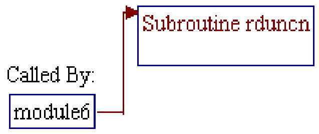
CALLS: None

CALLED BY: MODULE6() (in file MODULE6A.FOR)
Controls the type curve modules in generating type curve data.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name of the sub-program is declared. Header ".h" files are included.**

Note: Name of the sub-program is RDUNCN.

```
      SUBROUTINE RdUncn
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include'dimen.h'
      include'type3.h'
      include'type8.h'
```

Step 2: **Loop for pay grade is initialized.**

```
      Do I = 1, 3
```

Step 3: **Properties for dry shale reservoir (*KUnCon()=2*) are assigned.**

```
          If (KUnCon(I) .eq. 2) then
              If (PL(I).lt.1.)    PL(I)    = 1200.
              If (TDes(I).lt.1.)  TDes(I)  =  365.
              If (RhoMa(I).lt.1.) then
                      RhoMa(I) = 1360. * 2.35
              Else
                      RhoMa(I) = 1360. * RhoMa(I)
              End If
```

Step 4: **Properties for wet shale reservoir (*KUnCon()=3*) are assigned.**

```
          Else If (KUnCon(I) .eq. 3) then
              If (PL(I).lt.1.)    PL(I)    =  500.
              If (TDes(I).lt.1.)  TDes(I)  =  300.
              If (RhoMa(I).lt.1.) then
                      RhoMa(I) = 1360. * 2.3
              Else
                      RhoMa(I) = 1360. * RhoMa(I)
              End If
```

Step 5: **Properties for Appalachian coal reservoir (*ILoc()=0*) are assigned.**

```
          Else If (ILoc(I) .eq. 0) then
              If (PL(I).lt.1.)    PL(I)    =  220.
              If (TDes(I).lt.1.)  TDes(I)  =  365.
              If (RhoMa(I).lt.1.) then
```

```
                   RhoMa(I) = 1800.
            Else
                   RhoMa(I) = 1360. * RhoMa(I)
            End If
```

Step 6: **Properties for Warrior Basin coal reservoir (*ILoc()=1*) are assigned.**

```
        Else If (ILoc(I) .eq. 1) then
            If (PL(I).lt.1.)    PL(I)     =  150.
            If (TDes(I).lt.1.)  TDes(I)   =   10.
            If (RhoMa(I).lt.1.) then
                   RhoMa(I)  = 1800.
            Else
                   RhoMa(I) = 1360. * RhoMa(I)
            End If
```

Step 7: **Properties for Western coal reservoir (*ILoc()=2*) are assigned.**

```
        Else If (ILoc(I) .eq. 2) then
            If (PL(I).lt.1.)    PL(I)     =  300.
            If (TDes(I).lt.1.)  TDes(I)   =   10.
            If (RhoMa(I).lt.1.) then
                   RhoMa(I)  = 1800.
            Else
                   RhoMa(I) = 1360. * RhoMa(I)
            End If
        End If
```

Step 8: **Langmuir volume is calculated and pay grade loop is closed**

Note: *GasCon1()* is gas content (SCF/ton) and *Pinit()* is initial reservoir pressure (psia).

```
        VL(I) = GasCon1(I) * (1. + PL(I) / Pinit(I))
    End Do
```

Step 9: **The program control is returned back to the calling routine (sub-program MODULE6()) and the sub-program RDUNCN() is ended.**

```
    Return
    End
```

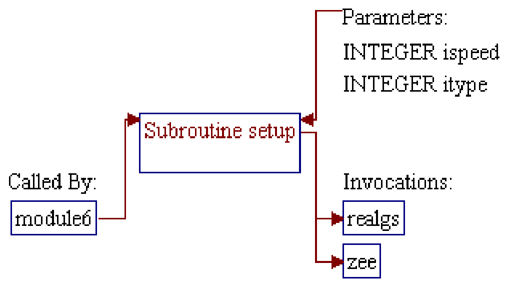
SUB-PROGRAM SETUP()

LOCATION:	MODULE6C.FOR
MAIN THEME:	This routine sets up tables of real gas potential (pseudo-pressure), gas viscosity, and gas Z-factor versus pressure for table lookup and calculates original gas in place.

CALLS:	REALGS() (in file MODULE6C.FOR)
	Calculates real gas potential (pseudo-pressure), gas viscosity, and gas compressibility factor (Z-factor) as functions of pressure.
	
	ZEE() (in file MODULE6C.FOR)
	Performs table look-up linear interpolation of Z-factor as a function of pressure.

CALLED BY:	MODULE6() (in file MODULE6A.FOR) Controls the type curve modules in generating type curve data.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included.**

Note: Name of the sub-program is SETUP() and the parameters passed to this sub-program are as follows:

- *Ispeed* Flag for standard formula (*ispeed=0*, higher accuracy) or simplified formula (*ispeed=1*, lower accuracy, faster)
- *IType* Reservoir type: 0=conventional, 1=water drive, 2=unconventional

```
      SUBROUTINE SetUp (ISpeed,IType)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type8.h'
      include 'type10.h'
```

Step 2: **Number or data points (*NArray*) and reservoir type (*IType*) are initialized.**

Note: *IType=1* is an indicator to skip refrac option.

```
      If (ISpeed .eq. 1) NArray = 99
      IType = 0
      If (((IMod(1,1)-5)*(IMod(2,1)-5)*(IMod(3,1)-5)).eq.0) then
              NArray = 99
              IType  = 1
      End If
```

Step 3: **Maximum pressure for tables is set.**

Note: 110% of the highest initial reservoir pressures is used as the maximum pressure for the table. This is to assure the ranges will be sufficient. This value should be at least 1000 psia.

```
      Pmax = Max (Pinit(1), Pinit(2), Pinit(3)) * 1.10
      If (Pmax .lt. 1000.) Pmax = 1000.
```

Step 4: **Sub-program REALGS() is invoked to calculate pseudo-pressure, gas viscosity, and gas Z-factor as functions of pressure.**

Note: This routine will generate arrays of pressure (*PreAry()*), pseudo-pressure (*PsiAry()*), gas viscosity (*VisAry()*), and gas Z-factor (*ZAry()*).

```
      Call REALGS (Pmax)
```

Step 5: **Original gas in place (*OGIP1()*) is calculated.**

Note: First obtain *OGIP1()* from free gas.

```
      Do J = 1, 3
         A  =  WSpace(J) * 43560.
         Pi =  Pinit(J)
         Zi =  Zee(Pi,NArray,PreAry,ZAry)
         OGIP1(J) = A * Thick(J) * Poros(J) * 520. / (Tem+460.) *
   +              ( 1. - Swi(J)) * Pi / Zi / 14.7 / 1000.
      End Do
```

Note: Add adsorbed gas from unconventional reservoirs to *OGIP1()*.

```
      If (((IMod(1,1)-6)*(IMod(2,1)-6)*(IMod(3,1)-6)).eq.0) then
          If ((KUnCon(1).eq.0) .or. (KUnCon(1).eq.2)) then
              IType = 2
          Else
              IType = 1
          End If
          Do J = 1, 3
              OGIP1(J) = OGIP1(J) + WSpace(J) * Thick(J) * RhoMa(J) *
   +                    VL(J) * Pinit(J) / (Pinit(J) + PL(J)) / 1000.
          End Do
      End If
```

Step 6: **The program control is returned back to the calling routine (sub-program MODULE6()) and the sub-program SETUP() is ended.**

```
      Return
      End
```

SUB-PROGRAM SETVALUE()

LOCATION: SETVALUE.FOR

MAIN THEME: This routine predicts the production years of developed reservoirs prior to the year 1993 to be used as the first time step in type curve routines.

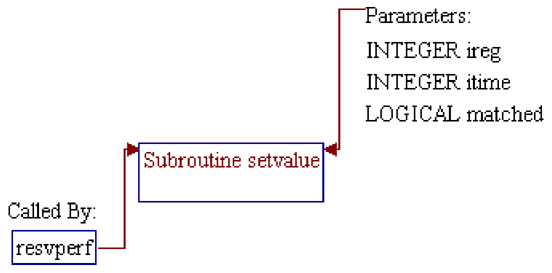
CALLS: None

CALLED BY: RESVPERF (in file RESVPERF.FOR)
Main program of Reservoir Performance Module.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included. Additional common blocks and local variables are declared.**

Note: Name of the sub-program is SETVALUE() and the parameters passed to this sub-program are as follows:

• *mathced*	Logical flag to indicate whether the reservoir has been matched (*matched=.TRUE.*) or not (*matched=.FALSE.*)
• *itime*	Time step number
• *ireg*	Region number (not currently used)

```
      subroutine setvalue(matched,itime,ireg)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'global.h'
      include 'cashflow.h'
      include 'field.h'
      include 'gsamvar.h'
      include 'welldata.h'
      include 'type_out.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
```

Note: Additional common blocks and local variables are declared.

```
      common /skinvalue/ skinfac(3)
      common /num_well/ nwella(3)
      logical matched
      integer itime,ireg
```

Step 2: **Time step number is increased by one.**

```
      itime=itime+1
```

Step 3: **Production history match is skipped if gas production rate in year 1993 (*gasprd93*) is less than or equal to 0.002 BCF.**

```
      if (gasprd93.le.0.002) then
      print *, '1993 Rate < 0.002 BCF/YR::Proceed Without History Check'
      goto 3001
      endif
```

Step 4: **Maximum methane production rate (*ratmax*) in year 1993 is calculated.**

```
      ratmax = gasprd93/(1.0-h2s-co2-n2)*1000000.0/365.0
```

Step 5: **Production years prior to the year 1993 is calculated and the value is stored in variable *tmematch*.**

Note: *tmematch* is calculated by dividing cumulative gas production up to the year 1993 (*cgpr93*) with the gas production in year 1993 (*gasprd93*).

```
      tmematch= cgpr93/gasprd93
```

Step 6: **First time step (*time(1)*) is set equal to *tmematch* and corrected with correction year (*corr_yr*).**

Note: Value of *corr_yr* is zero for undiscovered reservoirs and one for discovered reservoirs. Minimum value of the first time step is set to one year.

```
      time(1) = tmematch-corr_yr
      if(time(1).le.1.0) time(1)=1.0
```

Step 7: **Values of next time steps (up to 40 time steps) are uniformly set equal to one year.**

```
      do 106 i = 2,40
        time(i) = 1.0 + time(i-1)
 106  continue
```

Step 8: **Production years prior to drill infill wells for water drive reservoirs (*timchg*) is corrected.**

Note: Reservoir Module number for water drive reservoir is 5 (*module=5*). In this code, the *timchg* is corrected by shifting its value by *time(1)* years.

```
 3000 if (module.eq.5) timchg = time(1) + timchg
```

Step 9: **After performing production history calculation, the logical flag *matched* is set to *.TRUE.* (history matched has been done).**

```
      matched = .true.
```

Step 10: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program SETVALUE() is ended.**

```
 3001 return
      end
```

SUB-PROGRAM UNDWLSP()

LOCATION: READONE.FOR

MAIN THEME: This routine assigns undiscovered well spacing.

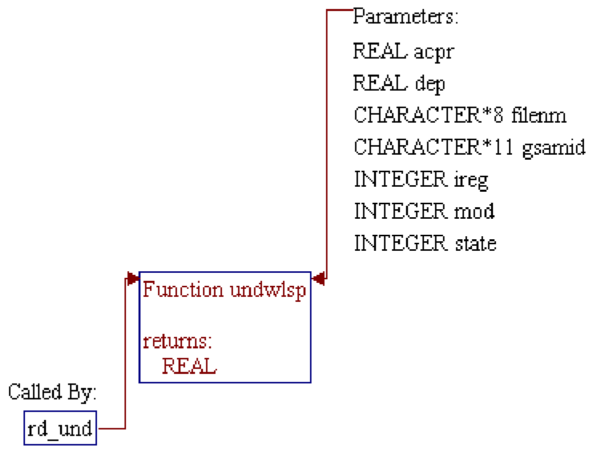
CALLS: None

CALLED BY: RD_UND() (in file READONE.FOR)
Reads one record from the one-line format .GSM file.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variables are declared.**

Note: Name of the sub-program is UNDWLSP() and the parameters passed to this sub-program are as follows:

Input Parameters:

• *IREG*	GSAM supply region code
• *MOD*	Reservoir Module number
• *DEP*	Depth of the reservoir (ft)
• *ACPR*	Drainage area of the reservoir (acres)
• *STATE*	4-digit State ID
• *DEP*	Depth of the reservoir (ft)
• *filenm*	Name of the [GSAM].GSM file in which reservoir is located
• *gsamid*	11-digit GSAM ID of the reservoir being processed

```
      FUNCTION UNDWLSP(IREG,MOD,DEP,ACPR,STATE,filenm,gsamid)
```

Note: Local variables are declared

```
      character *8 filenm
      character *11 gsamid
      integer state
```

Step 2: **At first, well spacing for all undiscovered reservoir is set to 320 acres.**

```
      wlsp=640./2.0
```

Step 3: **If GSAM supply region is less than or equal to “3” then well spacing is assigned to be 80 acres. In addition, if depth is less than 2000 ft then well spacing is only 40 acres.**

```
      if(ireg.le.3) then
        wlsp=160./2.0
        if(dep.lt.2000.) wlsp=80./2.0
```

Step 4: **If GSAM supply region is greater than “3” then if depth is less than 6000 ft then well spacing is set to 160 acres. In addition, if tight reservoirs are located in region number higher than “3” then, well spacing is set to 160 acres. Finally, if region number**

is greater than "3" and depth is less than 3000 ft, well spacing is set to 80 acres.

```
       else
         if(dep.lt.6000.) wlsp=320./2.0
         if(mod.eq.2)wlsp=320./2.0
         if(dep.lt.3000.)wlsp=160./2.0
       endif
```

Step 5: **If the above calculation provided 1.5 times the well spacing greater than the area of the reservoir, then well spacing (*wlsp*) is set to the reservoir area (*acpr*) itself.**

```
       if(wlsp*1.5.gt.acpr)wlsp=acpr
```

Step 6: **The following section assigned well spacing for coalbed methane wells (*mod* = *6*). For GSAM supply regions less than and equal to 8, then well spacing is set to 160 acres. For other regions it is set to 80 acres.**

```
       if(mod.eq.6) then
         if (ireg.ge.8) then
           wlsp = 160.
          else
           wlsp=80.
          endif
        endif
```

Step 7: **Finally, the adjustments are done as follows. If GSAM supply region is less than 13, then well spacing is half of well spacing calculated earlier.**

```
       if(ireg.lt.13)wlsp=wlsp/2.0
```

Step 8: **If GSAM supply region is equal to 10 and resource type is 2 (i.e. tight) then well spacing equals to double the well spacing calculated earlier.**

```
       if (ireg.eq.10.and.mod.eq.2) wlsp = wlsp*2.0
```

Step 9: **The function variable *undwlsp* is set to *wlsp* variable.**

```
       undwlsp=wlsp
```

Step 10: **For Canadian reservoirs, all undiscovered reservoirs are at well spacing of 80 acres.**

```
       if (gsamid(1:2).eq.'22'.or.gsamid(1:2).eq.'23'.or.
      @    gsamid(1:2).eq.'24') then
         undwlsp =  80.0
       endif
```

Step 11: **The following section is set for reservoirs located in Appalachia. For Appalachian reservoirs (*GSAMID(1:2) = '01'*) and discovered producing section, well spacing is defaulted to 80 acres first. Then, different set of data is gathered together to create a colorful portrayal of well spacing for undiscovered reservoirs as shown below.**

```
        if (gsamid(1:2).eq.'01'.and. gsamid(3:3).eq.'3') then
          undwlsp = 80.0
          if (state.eq.16) undwlsp = 50
          if (state.eq.31) undwlsp = 90
          if (state.eq.34) undwlsp = 110
          if (state.eq.37) undwlsp = 70
          if (state.eq.41) undwlsp = 120
          if (state.eq.47) undwlsp = 30
          if (state.eq.45) undwlsp = 40
        endif
```

Step 12: **Program control is returned back to the calling routines (sub-program RD_UND()) and the sub-program UNDWLSP() is ended.**

```
         Return
         End
```

SUB-PROGRAM CALCOF()

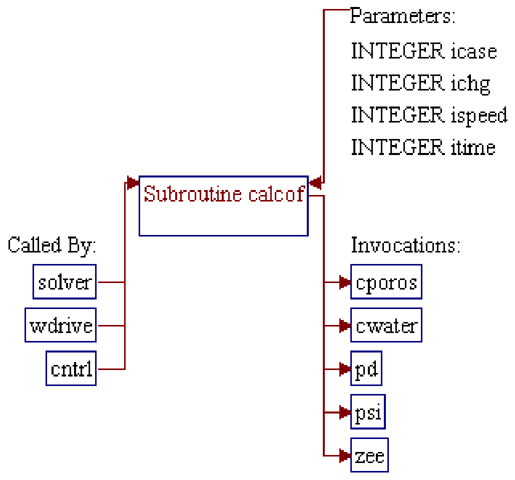
LOCATION:	MODULE6B.FOR
MAIN THEME:	This routine calculates open flow potential.

CALLS:	CPOROS() (in file MODULE6D.FOR)
	Computes pore volume compressibility using a curve fit to Hall's correlation.
	
	CWATER() (in file MODULE6D.FOR)
	Calculates water compressibility using OSIF's correlation (SPE Reservoir Engineering, Feb. 1988).
	
	PD() (in file MODULE6B.FOR)
	Calculates dimensionless pressure functions for different reservoir syustems.
	
	PSI() (in file MODULE6B.FOR)
	Performs table look-up of real gas potential (psudo-pressure) given pressure.
	
	ZEE() (in file MODULE6C.FOR)
	Performs table look-up linear interpolation of Z-factor as a function of pressure.

CALLED BY:	SOLVER() (in file MODULE6C.FOR)
	Solves for flow rates and pressures within specified time step.
	
	WDRIVE() (in file MODULE6C.FOR)
	Computes performances of water drive reservoirs using water influx material balance.
	
	CNTRL() (in file MODULE6D.FOR)
	Initializes minimum pressures, maximum rates, skins, infill wells ON/OFF, etc.

READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variable are declared.**

Note: Name of the sub-program is CALCOF() and the parameters passed to this sub-program are as follows:

• *ITime*	Time step number
• *ICase*	Case number: 1=primary, 2=automatic refrac, 3=automatic infill one time.
• *IChg*	Flag to indicate whether development type change has taken place: 0=primary wells only, 1=one set of infills, 2=two sets of infills (not currently used), 3= primary wells after restimulation
• *ISpeed*	Flag for standard formula (*ispeed=0*, higher accuracy) or simplified formula (*ispeed=1*, lower accuracy, faster)

```
      SUBROUTINE CALCOF (ITime,  ICase,  IChg,   ISpeed)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
```

Note: Local variable for dimensionless pressures is defined.

```
      Dimension Pdcon(3,3)
```

Step 2: **Pay grade loop is initialized. Flow potential is calculated separately for each reservoir pay grade.**

```
      Do I = 1, 3
```

Step 3: **Dimensionless time based on drainage area (*Tda*) and constant term of dimensionless pressure (*PCon*) are calculated.**

Note: Maximum pressure (*P1*) is obtained either from initial pressure (*Pinit()*) (for the first time step) or from average pressure (*PreAvg()*) of the previous time step (for time step greater than 1). Minimum pressure (*P2*) is atmospheric pressure (*14.7* psia). *PAvg*, an average pressure between *P1* and *P2*, is calculated to be used for water compressibility calculation. Note that the open flow potential calculation is skipped if *P1* for the corresponding pay grade is zero or negative.

```
                  P1     = Pinit(I)
                  If (ITime.gt.1) P1 = PreAvg(I,ITime-1)
                  if(p1.le.0.0) go to 1109
                  P2     = 14.7
                  PAvg   = (P1 + P2) / 2.
```

Note: The following codes convert units of well spacing (*WSpace()*) from acres to square feet and calculate water and pore volume compressibilities using sub-programs CWATER() and CPOROS(), respectively.

```
                  A      = WSpace(I) * 43560.
                  Cw     = Cwater(Pavg, Tem, Salin(I))
                  Cp     = Cporos(Poros(I))
```

Note: Sub-programs ZEE() and PSI() are invoked to calculate Z-factors and pseudo-pressure at pressures *P1* and *P2*.

```
                  Z1     = Zee(P1,NArray,PreAry,ZAry)
                  Z2     = Zee(P2,NArray,PreAry,ZAry)
                  Psi1   = Psi(P1,NArray,PreAry,PsiAry)
                  Psi2   = Psi(P2,NArray,PreAry,PsiAry)
```

Note: Dimensionless time based on drainage area *(Tda)* and constant term of dimensionless pressure (*PCon*) are calculated. *CMuEff* is effective compressibility-viscosity product.

```
                  Ct     =(Cw * Swi(I) + Cp)/(1.-Swi(I))
                  CMuEff = 2. * (1.-Swi(I)) / (Psi1-Psi2) *
     +                        (P1/Z1 - P2/Z2 * (1. - Ct*(P1-P2)))
                  Tda    = 0.006328*Perm(I)*365. / (Poros(I)*CMuEff*A)
                  PCon   = 1422.*(Tem+460.)/(Perm(I)*Thick(I))
```

Step 4: **Dimensionless pressures at the wellbore *(Pdw)*, at the corner *(PdCorn)*, and at the second infill *(PdEdge)* are calculated.**

Note: Development type loop is initialized.

```
                 Do J = 1, 3
```

Note: Get module type (*Module*) and effective wellbore radius (*Rweff*).

```
                         Module = IMod(I,J)
                         Rweff  = Rw  (I,J)
```

Note: Recalculate effective wellbore radius (*Rweff*) for fractured reservoirs (modules 2 and 4). For vertical well with infinite conductivity fracture (*Fcd < 0*) or horizontal well (*JTyp=1*) the dimensionless fracture conductivity (*Fcd*) is set to 1E6. Note that the RP Module treats the horizontal well as an infinite conductivity fracture.

```
                         If ((Module.eq.2).or.(Module.eq.4)) then
                                 DLen  = HalfLn(I,J)
                                 Fcd   = Cond(I,J)/(Perm(I)*DLen)
                                 If (Fcd .le. 0.) Fcd = 100000.
                                 SHor  = 0.
                                 Rweff = DLen
                                 If (JTyp(I,J) .eq. 1) then
                                         DLen  = HorLen(I,J)
                                         Fcd   = 100000.
                                         Rweff = DLen/2.
                                         Ratio = Sqrt(PermV(I)/Perm(I))
                                         Rwd   = Rw(I,J)/Thick(I)*Ratio
                                         SHor  = -2.*Thick(I)/DLen/Ratio*
     +                                        Log(2.*Asin(3.1415926*Rwd))
                                 End If
                         End If
```

Note: Sub-program PD() is invoked to calculate dimensionless pressures. Some parameters such as dimensionless area factor (*ARw*), Warren and Root porosity-compressibility ratio (*Omega*), and Warren and Root interporosity flow parameter (*DLam*) are calculated and passed to the PD() sub-program. The results are stored in array variable *Pdcon()* and the development loop *J* is closed.

```
                         ARw = Sqrt(A)/Rweff
                         If ((Module.eq.3).or.(Module.eq.4)) then
                                 Omega = PorMa(I)/Poros(I)
                                 DLam  = 12. * PermMa(I)/Perm(I) *
     +                                   (Rweff/FrcSpc(I))**2
                         End If
                         Call PD(Tda,Module,ARw,Fcd,SHor,Omega,DLam,
     +                           ISpeed,Pdw,PdCorn,PdEdge,IErr)
                         If (J .eq. 1) then
                                 Pdcon(J,1) = Pdw
                                 Pdcon(J,2) = PdCorn
                                 Pdcon(J,3) = PdEdge
                         Else If (J .eq. 2) then
                                 Pdcon(J,1) = PdCorn
                                 Pdcon(J,2) = Pdw
                                 Pdcon(J,3) = PdEdge
                         Else
                                 Pdcon(J,1) = PdEdge
                                 Pdcon(J,2) = PdEdge
                                 Pdcon(J,3) = Pdw
                         End If
                 End Do
```

Step 5: **Absolute open flow potential (*CAOF()*) is calculated.**

Note: If the development case has not yet changed *(IChg=0)* or the case is for primary well or automatic refracturing *(ICase=1* or *2*), open flow potentials for infills (*CAOF(...,2,...)* and *CAOF(...,3,...)*) are set to zeros. If the change has taken place from primay to infill one time (*IChg=1*) the *CAOF(...,2,...)* is also calculated.

```
                  If ((ICase.eq.1).or.(ICase.eq.2).or.(IChg.eq.0)) then
                         S = Skin(I,1,1)
                         If (IChg .eq. 3) S = Skin(I,1,2)
                         CAOF(I,1,ITime) = Psi1 / PCon / (Pdcon(1,1)+S)
                         CAOF(I,2,ITime) = 0.
                         CAOF(I,3,ITime) = 0.
                  Else If (IChg.eq.1) then
                         A11 = PCon * (Pdcon(1,1)+Skin(I,1,1))
                         A12 = PCon *  Pdcon(1,2)
                         A21 = PCon *  Pdcon(2,1)
                         A22 = PCon * (Pdcon(2,2)+Skin(I,2,1))
                         CAOF(I,1,ITime) = (Psi1 * A22 - Psi1 * A12) /
     +                                     (A11  * A22 - A21  * A12)
                         CAOF(I,2,ITime) = (Psi1 * A11 - Psi1 * A21) /
     +                                     (A11  * A22 - A21  * A12)
                         CAOF(I,3,ITime) = 0.
                  End If
```

Note: All open flow potentials are set to zero if the reservoir unit is shut in (*KShut > 0*).

```
                  If(KShut(I).gt.0) then
                         CAOF(I,1,ITime) = 0.
                         CAOF(I,2,ITime) = 0.
                         CAOF(I,3,ITime) = 0.
                  End If
```

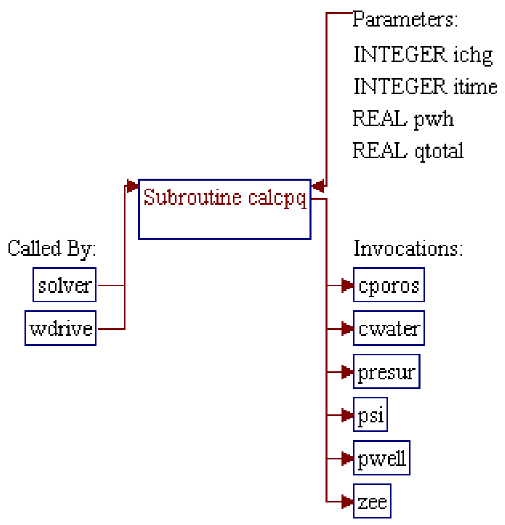
Step 6: **Pay grade loop is closed, program control is returned back to the calling routines (sub-program SOLVER(), WDRIVE(), or CNTRL()) and the sub-program CALCOF() is ended.**

```
 1109      End Do
        Return
        End
```

SUB-PROGRAM CALQPQ()

LOCATION:	MODULE6D.FOR
MAIN THEME:	This routine computes wellhead and bottom hole pressures after rates have been determined.
CALLS:	CPOROS() (in file MODULE6D.FOR) Computes pore volume compressibility using a curve fit to Hall's correlation.
	CWATER() (in file MODULE6D.FOR) Calculates water compressibility using OSIF's correlation (SPE Reservoir Engineering, Feb. 1988).
	PRESUR() (in file MODULE6B.FOR) Performs inverse table look-up of pressure given real gas potential (psudo-pressure).
	PSI() (in file MODULE6B.FOR) Performs table look-up of real gas potential (psudo-pressure) given pressure.
	PWELL() (in file MODULE6B.FOR) Calculates bottomhole pressure, wellhead pressure, or flow rate based on the difference between well head pressure and bottom hole pressure of the well using Smith's formula.
	ZEE() (in file MODULE6C.FOR) Performs table look-up linear interpolation of Z-factor as a function of pressure.
CALLED BY:	SOLVER() (in file MODULE6C.FOR) Solves for flow rates and pressures within specified time step.
	WDRIVE() (in file MODULE6C.FOR) Computes performances of water drive reservoirs using water influx material balance.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included.**

Note: Name of the sub-program is CALCPQ() and the parameters passed to this sub-program are as follows:

• *Pwh*	wellhead pressure (psia)
• *QTotal*	Total production from field (MCFD)
• *ITime*	Time step number
• *IChg*	Flag to indicate whether development type change has taken place: 0=primary wells only, 1=one set of infills, 2=two sets of infills (not currently used), 3= primary wells after restimulation

```
      SUBROUTINE CalcPQ (Pwh, QTotal, ITime, IChg)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type10.h'
```

Step 2: **Time step size *DT* is set.**

Note: Array variable *Time()* stores time data (e.g. 1, 2, 3,...) to be analyzed. The time step size is calculated by subtracting years from two consecutive data points. The size of the first time step is equal to the number of years in the first data point.

```
      DT = Time(ITime)
      If (ITime.gt.1) DT = DT - Time(ITime-1)
```

Step 3: **Stop production (*KShut=1*) if total flow rate (*Q*) or maximum flow rate based on maximum recovery efficiency (*QMax3*) is too low.**

Note: Maximum recovery efficiency (*REMax*) for gas expansion reservoirs (*IMod()<>5*) is based on P/Z at minimum pressure (*P1/Z1*). For water drive reservoirs (*IMod()=5*), maximum recovery efficiency is based on trapping gas at the average reservoir pressure behind the encroaching water front and minimum possible pressure in the unencroached reservoir. Sub-program ZEE() is used for gas Z-factor calculation.

```
      Do J = 1, 3
          Pi   = Pinit(J)
          Zi   = Zee(Pi,NArray,PreAry,ZAry)
          P1   = PreMin
          Z1   = Zee(P1,NArray,PreAry,ZAry)
          P2   = PreAvg(J,ITime)
          Z2   = Zee(P1,NArray,PreAry,ZAry)
          If (IMod(J,1).ne.5) then
              REMax = 1.-(P1/Z1)/(Pi/Zi)
          Else
              WeD1 = WeD(J) / (1.-Swi(J)-SgTrap(J))
              REMax = 1.-(P1/Z1)/(Pi/Zi)*(1.-Swi(J))*(1.-WeD1)
     +                  -(P2/Z2)/(Pi/Zi)*  SgTrap(J)*    WeD1
          End If
          QMax3 = (OGIP1(J)*REMax-CumGas(J,1,ITime)-CumGas(J,2,ITime)-
     +                2.* CumGas(J,3,ITime)) / (365.*DT)
          Q = Qg(J,1,ITime) + Qg(J,2,ITime) + Qg(J,3,ITime) * 2.
          If ((Q.lt.10.).or.(QMax3.lt.10.)) KShut(J) = 1
      End Do
```

Step 4: **Changes in gas flow rates (*DQ()*) and cumulative productions (*CumGas()*) for current time step are calculated.**

```
      Do I = 1, 3
              Do J = 1, 3
                      If (ITime .eq. 1) then
                          DQ(I,J,ITime) = Qg(I,J,ITime)
                          CumGas(I,J,ITime)=Qg(I,J,ITime)*365.*Time(1)
                      Else
                          DQ(I,J,ITime)=Qg(I,J,ITime)-Qg(I,J,ITime-1)
                          CumGas(I,J,ITime) = CumGas(I,J,ITime-1)+
     +                        Qg(I,J,ITime)*365.*
     +                       (Time(ITime)-Time(ITime-1))
                      End If
              End Do
      End Do
```

Step 5: **Wellhead pressure and bottomhole pressure are calculated.**

Note: Sub-program PSI() is invoked to convert initial pressure (*Pinit()*) into pseudo-pressure (real gas potential). The pseudo-pressure is updated with pseudo-pressure drops due to productions, the terms with *C(p1,p2,p3)*, where *p1* is for pay grade, *p2* is for primary or infill wells, and *p3* is interference term for other wells (1=corner of primary, 2=corner of infill, 3=edge).

```
         Do I = 1, 3
                Do J = 1, 3
                        P = Pinit(I)
                        RGP = Psi(P,NArray,PreAry,PsiAry)-DPsi(I,J)
                        Q   = Qg(I,J,ITime)
                        S = Skin(I,J,1)
                        If ((IChg.eq.3) .and. (J.eq.1)) S = Skin(I,1,2)
                        RGP = RGP - PsiCon(I) * ( S * Q +
     +                              DQ(I,1,ITime) * C(I,J,1) +
     +                              DQ(I,2,ITime) * C(I,J,2) +
     +                              DQ(I,3,ITime) * C(I,J,3) * 2. )
```

Note: Sub-program PRESUR() is invoked to calculate bottomhole pressure based on the calculated pseudo-pressure (*RGP*). Sub-program PWELL() is invoked to calculate wellhead pressure given the calculated bottomhole pressure (*Pbh*) and gas flow rate (*Qg(),Q*).

```
                        KTyp = KUnCon(I)
                        Pbh = Presur(RGP,NArray,PreAry,PsiAry)
                        Q = Qg(I,J,ITime)
                        Dep = Depth1(I)
                        Call PWELL(Pwh,Pbh,Q,Deriv,Dep,2,IErr,KTyp,I)
                        Prwh(I,J,ITime) = Pwh
                        Prbh(I,J,ITime) = Pbh
                End Do
         End Do
```

Step 6: **Average reservoir pressure (*PreAvg()*) for conventional reservoirs (not water drive reservoirs) is determined iteratively using Bi-section method.**

Note: The water drive reservoirs are skipped because the average pressure already determined in calling routine WDRIVE() using water influx material balance computations.

```
         If (IMod(1,1).ne.5) then
```

Note: Calculate objective functions at initial pressure (*Fn*) and at the first entry of pressure table *PreAry()* (*F*). *PreAry()* is a table of pressure from zero to about 10% higher than the highest initial pressures. This table consists of *NArray* number of data (typically 99 points).

```
             Do I = 1, 3
                Cp   = Cporos(Poros(I))
                Pi   = Pinit(I)
                Zi   = Zee(Pi,NArray,PreAry,ZAry)
                Gp   = CumGas(I,1,ITime) + CumGas(I,2,ITime) +
     +                   CumGas(I,3,ITime) * 2.
                Fn   = Pinit(I)/Zi * (1. - Gp/OGIP1(I))
                I1   = 1
                P    = PreAry(I1)
                Z    = ZAry(I1)
                Pavg = (Pinit(I) + P) / 2.
                Cw   = Cwater(Pavg, Tem, Salin(I))
                Cwp  = (Cw*Swi(I)+Cp)/(1.-Swi(I))
                F    = P/Z*(1.-Cwp*(Pi-P)-WeD(I))
```

Note: If reservoir pressure is less than the first entry in the pressure table (a condition when *F>Fn*), the average reservoir pressure is set/calculated directly. *PreAvg()* is set to 14.7 psia if objective function at initial pressure (*Fn*) is negative. Otherwise (*Fn*>0), *PreAvg()* is calculated based on material balance at initial pressure.

```
                If (F .gt. Fn) then
                        A = Cwp
                        B = 1.-Cwp*Pi-WeD(I)+Fn*(1.-Z)/P
                        Disc = B*B + 4*A*Fn
                        If (Fn.gt.0.) then
                                PreAvg(I,ITime) = 2.*Fn/(B+Sqrt(Disc))
                        Else
                                PreAvg(I,ITime) = 14.7
                        End If
```

Note: If reservoir pressure is at least equal to the first entry in the pressure table (a condition when *F >= Fn*), the average reservoir pressure is calculated iteratively. In this procedure, a maximum of *Ln(NArray)/Ln(2)+1* Bi-section iterations (about 7 iterations for *NArray* of 99 data points) is performed to get a range of pressure for material balance calculation. The iteration is started with the highest pressure range (a range between the first and last entries of the pressure table). The iteration process is stopped if the two pressures in the table designated by pointers *I0* and *I2* are next to each other. At the end of the iteration, average reservoir pressure is calculated based on material balance.

```
                Else
                        I0   = 1
                        I2   = NArray
                        DN   = Float(NArray)
                        D    = Log(DN)/Log(2.) + 1.
                        JMax = Int(D)
                        Do J = 1, JMax
                                If (I2-I0-1 .gt. 0) then
                                        I1 = (I2+I0)/2
                                        P  = PreAry(I1)
                                        Z  = ZAry(I1)
                                        Pavg =(Pinit(I) + P) / 2.
                                        Cw = Cwater(Pavg, Tem, Salin(I))
                                        Cwp = (Cw*Swi(I)+Cp)/(1.-Swi(I))
                                        F = P/Z*(1.-Cwp*(Pi-P)-WeD(I))
                                        If (F .ge. Fn) then
                                                I2 = I1
                                        Else
                                                I0 = I1
                                        End If
                                End If
                        End Do
                        Cwp = (Cw*Swi(I)+Cp)/(1.-Swi(I))
                        dZdP=(ZAry(I2)-ZAry(I0))/(PreAry(I2)-PreAry(I0))
```

```
                       U   = (ZAry(I0)-dZdP*PreAry(I0)) * Fn
                       B   = 1 - Cwp * Pi - WeD(I) – Fn * dZdP
                       Disc= B*B + 4*Cwp*U
                       PreAvg(I,ITime)=2.* U / ( B + Sqrt(Disc))
               End If
           End Do
        End If
```

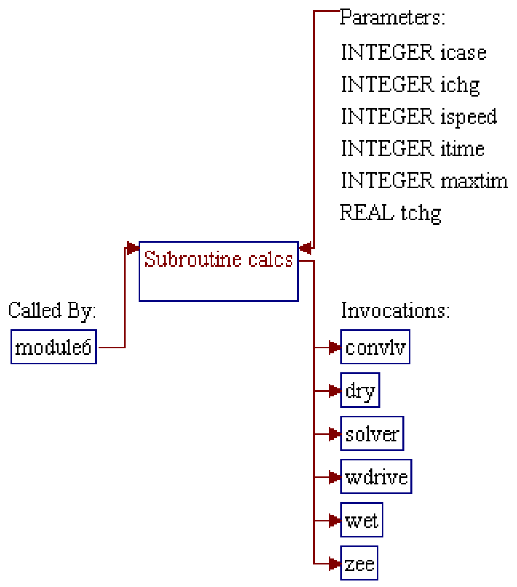
Step 7: **The program control is returned back to the calling routine (sub-program SOLVER() or WDRIVE()) and the sub-program CALCPQ() is ended.**

```
         Return
         End
```

SUB-PROGRAM CALCS()

LOCATION:	MODULE6D.FOR
MAIN THEME:	This routine performs numerical convolution and solves for pressure and flow rates at each time step to generate type curves. Initially assume average reservoir pressure at the end of the time step is equal to pressure at the beginning. Iteration process is performed to get average reservoir pressure.
CALLS:	CONVLV () (in file MODULE6C.FOR) Performs numerical convolution to determine pressure drop caused by previous production.
	DRY() (in file MODULE6A.FOR) Controls sub-routine DRY() to calculate gas flow rates for dry coal and dry shale reservoirs.
	SOLVER() (in file MODULE6C.FOR) Solves for flow rates and pressures within specified time step.
	WDRIVE() (in file MODULE6C.FOR) Computes performances of water drive reservoirs using water influx material balance.
	WET() (in file MODULE6A.FOR) Computes performances of wet coal and shale reservoirs using material balance approach.
	ZEE() (in file MODULE6C.FOR) Performs table look-up linear interpolation of Z-factor as a function of pressure.
CALLED BY:	MODULE6() (in file MODULE6A.FOR) Controls the type curve modules in generating type curve data.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included. Additional common block and local variables are declared.**

Note: Name of the sub-program is CALCS() and the parameters passed to this sub-program are as follows:

• *ITime*	Time step number
• *ICase*	Case number: 1=primary, 2=automatic refrac, 3=automatic infill one time, 4=automatic infill twice (not yet implemented).
• *IChg*	Flag to indicate whether development type change has taken place: 0=primary wells only, 1=one set of infills, 2=two sets of infills (not currently used), 3= primary wells after restimulation
• *TChg*	Time at which automatic change in development type occurs (automatic infill or refrac)
• *ISpeed*	Flag for standard formula (*ISpeed=0*, higher accuracy) or simplified formula (*ISpeed=1*, lower accuracy, faster)
• *MaxTim*	Maximum number of time steps

```
      SUBROUTINE Calcs (ITime,ICase,IChg,TChg,ISpeed,MaxTim)
```

Note: Header .h files which declare global variables and common blocks are included.

```
	include 'dimen.h'
	include 'type2.h'
	include 'type5.h'
	include 'type10.h'
	include 'type3.h'
	include 'type4.h'
	include 'type8.h'
```

Note: Additional common block and local variables are declared.

```
      common /stchg/iwin_yr
      Dimension PGuess(3), JShut(3)
```

Step 2: **Working variables for convergence flag (*JTol*) is initialized. Flag of development change (*IChg0*) and time of development type change (*TChg0*) are stored to working variables.**

Note: *JTol* tells whether convergence has occured (0=no, 1=yes).

```
JTol  = 0
TChg0 = TChg
IChg0 = IChg
```

Step 3: **Previous time step data (average reservoir pressure (*PreAvg()*) and cumulative gas production (*CumGas()*)) are returned if all pay grades are already shut in (*ΣKShut()=3*).**

```
If ((KShut(1)+KShut(2)+KShut(3)).eq.3) then
        Do I = 1, 3
                PreAvg(I,ITime)   = PreAvg(I,ITime-1)
                CumGas(I,1,ITime) = CumGas(I,1,ITime-1)
                CumGas(I,2,ITime) = CumGas(I,2,ITime-1)
                CumGas(I,3,ITime) = CumGas(I,3,ITime-1)
        End Do
        Return
End If
```

Step 4: **For Reservoir Module 5 (*IMod()=5*), sub-program WDRIVE() is invoked to perform calculations for water drive reservoirs.**

Note: If one pay grade is water drive, all other pay grades are assumed to be water drives. To indicate convergence, flag *JTol* is set to 1 and pressure difference (*DP*) is set to zero. *DP* is pressure difference between two consecutive iteration levels.

```
If (((IMod(1,1)-5)*(IMod(2,1)-5)*(IMod(3,1)-5)).eq.0) then
        Call WDRIVE (ITime,ICase,MaxTim)
        JTol = 1
        DP = 0.
```

Step 5: **For Reservoir Module 6 (*IMod()=6*), sub-program DRY() or WET() is invoked to perform calculations for unconventional (coal and shale) reservoirs.**

Note: If one pay grade is unconventional, all other pay grades are assumed to be unconventional reservoirs. Flag *KUnCon()* is used to indicate dry coal/shale (*KUnCon()=0* or *2*) or wet coal/shale (*KUnCon()=1* or *3*). Flag *JTol* is set to 1 and pressure difference (*DP*) is set to zero to indicate that convergence has achieved.

```
         Else If (((IMod(1,1)-6)*(IMod(2,1)-6)*(IMod(3,1)-6)).eq.0) then
                 If ((KUnCon(1).eq.0) .or. (KUnCon(1).eq.2)) then
                         Call DRY (ITime,ICase,IChg,MaxTim,IOF)
                 Else
                         Call WET (ITime,ICase,IChg,MaxTim,IOF)
                 End If
                 JTol = 1
                 DP = 0.
```

Step 6: **For other reservoir systems (Reservoir Modules 1 through 4), semi-analytical real gas potential (pseudo-pressure) approach is utilized. The first step is to set pressure tolerance (*PTol*) and maximum number of iterations (*JMax*) based on speedup option (*ISpeed*).**

Note: *PTol* and *Jmax* are used in iterative procedure to solve for average reservoir pressure. The magnitudes of these parameters are set depending on the speedup flag (*ISpeed*). In the base case (*ISpeed=0*), pressure tolerance and maximum number of iterations are 2 psi and 13 iterations, respectively. If speedup option is requested (*ISpeed=1*), coarser pressure tolerance (*PTol=25* psi) and smaller number of iterations (*JMax=3*) are utilized to get the results (less accurate results) with lower CPU time.

```
         Else
             If (ISpeed .eq. 1) then
                 PTol =  25.
                 JMax =  3
             Else
                 PTol =  2.
                 JMax =  13
             End If
```

Step 7: **Working variables and initial guesses are set.**

Note: Flags for shut-in (*KShut()*) and development change (*IChg*) are stored to working variables *JShut()* and *KChg*, respectively. For the first time step, initial reservoir pressures (*Pinit()*) are used as initial guess pressures (*PGuess()*). For other time steps, average reservoir pressures from previous time step (*PreAvg(I,ITime-1)*) are used as the initial guess pressures.

```
             Do I = 1, 3
                 JShut(I)  = KShut(I)
                 PGuess(I) = Pinit(I)
                 If (ITime.gt.1) PGuess(I) = PreAvg(I,ITime-1)
                 PreAvg(I,ITime) = PGuess(I)
             End Do
             KChg  = IChg
```

Step 8: **Successive-Substitution iterative method is performed to solve for average reservoir pressure.**

Note: The iterative process is performed with maximum number of iterations *JMax* and pressure tolerance *PTol* (see Step 6). The following codes initialize iteration loop and recall the shut-in flags.

```
      Do J = 1, JMax
          If (JTol.eq.0) then
                  KShut(1) = JShut(1)
                  KShut(2) = JShut(2)
                  KShut(3) = JShut(3)
```

Note: Sub-program CONVLV() is invoked to determine pressure drops caused by previous production (numerical convolution or superposition in time).

```
                  Call CONVLV(ITime, J, ISpeed)
```

Note: Time of development type change (*TChg*) and flag of development type change (*IChg*) are recalled. Sub-program SOLVER() is then invoked to solve for flow rates and pressures for the current time step.

```
                  TChg = TChg0
                  IChg = IChg0
                  Call SOLVER(ITime, ICase, IChg, TChg, ISpeed)
```

Note: Deviations between guessed and calculated average reservoir pressures (*DP1*, *DP2*, and *DP3*) are calculated. The maximum deviation is stored in variable *DP*.

```
                  DP1 = PGuess(1)-PreAvg(1,ITime)
                  DP2 = PGuess(2)-PreAvg(2,ITime)
                  DP3 = PGuess(3)-PreAvg(3,ITime)
                  DP  = Max(Abs(DP1),Abs(DP2),Abs(DP3))
```

Note: Convergence check is performed. *IChk* is used to check whether the calculated pressure is higher than the guessed pressure (*IChk=1*). The convergence is achieved *(JTol=1)* if the maximum deviation is at most equal to pressure tolerance (*DP<=PTol*), the calculated pressure is not increased (*IChk=0*), and number of iterations is greater than 1 (*J>1*). The iteration is repeated until the convergence is achieved.

```
                         IChk = 0
                         Do K = 1, 3
                            If (PGuess(K).gt.PInit(K)) IChk = 1
                         End Do
                         If ((DP.le.PTol).and.(IChk.eq.0).and.(J.gt.1))
     +                           JTol = 1
                 End If
             End Do
             IChg   = KChg
         End If
```

Step 9: **Average reservoir pressure is set to the value at previous time step if gas flow rate is too low. Water influx of previous time step is set to the calculated value.**

```
         Do I=1,3
             If (Qg(I,1,ITime).lt.1.) PreAvg(I,ITime)=PreAvg(I,ITime-1)
             WePrev(I) = We(I)
         End Do
```

Step 10: **The program control is returned back to the calling routine (sub-program MODULE6()) and the sub-program CALCS() is ended.**

```
         Return
         End
```

SUB-PROGRAM CONVLV()

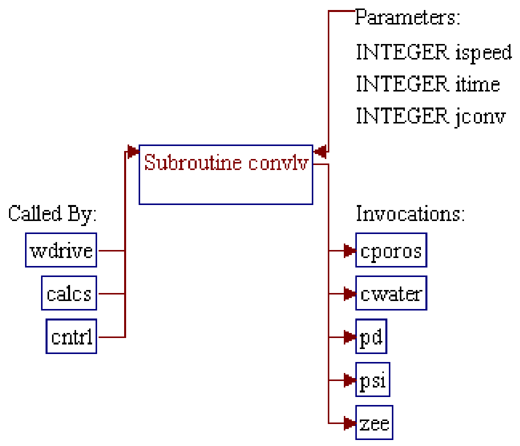
LOCATION:	MODULE6D.FOR
MAIN THEME:	This routine calculates pressure drops caused by previous productions using numerical convolution (superposition in time principle).

CALLS:	CPOROS() (in file MODULE6D.FOR)
	Computes pore volume compressibility using a curve fit to Hall's correlation.
	
	CWATER() (in file MODULE6D.FOR)
	Calculates water compressibility using OSIF's correlation (SPE Reservoir Engineering, Feb. 1988).
	
	PD() (in file MODULE6B.FOR)
	Calculates dimensionless pressure functions for different reservoir systems.
	
	PSI() (in file MODULE6B.FOR)
	Performs table look-up of real gas potential (psudo-pressure) given pressure.
	
	ZEE() (in file MODULE6C.FOR)
	Performs table look-up linear interpolation of Z-factor as a function of pressure.

CALLED BY:	WDRIVE() (in file MODULE6C.FOR)
	Computes performances of water drive reservoirs using water influx material balance.
	
	CALCS() (in file MODULE6D.FOR)
	Performs numerical convolution and solves for pressure and flow rates at each time step to generate type curves.
	
	CNTRL() (in file MODULE6D.FOR)
	Initializes minimum pressures, maximum rates, skins, infill wells ON/OFF, etc.

READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included.**

Note: Name of the sub-program is CONVLV() and the parameters passed to this sub-program are as follows:

• *ITime*	Time step number
• *JConv*	Iteration level (it is used to tell whether this is the first iteration or not)
• *ISpeed*	Flag for standard formula (*ISpeed=0*, higher accuracy) or simplified formula (*ISpeed=1*, lower accuracy, faster)

```
      SUBROUTINE Convlv (ITime, JConv, ISpeed)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
```

Step 2: **Loop for pay grads is initialized.**

```
      Do J = 1, 3
```

Step 3: **Sub-programs ZEE() and PSI() are invoked to calculate gas Z-factor and pseudo-pressure at initial reservoir pressure.**

```
              Pi   = Pinit(J)
              Zi   = Zee(Pi,NArray,PreAry,ZAry)
              Psii = Psi(Pi,NArray,PreAry,PsiAry)
```

Step 4: **Average reservoir pressure (*PRes*) is set based on iteration level (*JConv*).**

Note: For the first iteration (*JConv=1*), the average reservoir pressure (*PreAvg()*) is assumed to be 100 psi lower than the initial reservoir pressure (if this is the first time step), or it is set to the average reservoir pressure at previous time, otherwise. For iteration level greater than 1 (*JConv>1*) use the available average reservoir pressure.

```
              If (JConv.eq.1) then
                     PRes = Pi - 100.
                     If (ITime.gt.1) PRes = PreAvg(J,ITime-1)
                     PreAvg(J,ITime) = PRes
              Else
                     PRes = PreAvg(J,ITime)
              End If
```

Step 5: **Sub-programs ZEE() and PSI() are invoked to calculate gas *Z*-factor and pseudo-pressure at average reservoir pressure.**

```
              Z       =  Zee(PRes,NArray,PreAry,ZAry)
              PsiRes  =  Psi(PRes,NArray,PreAry,PsiAry)
```

Step 6: **Sub-programs CWATER() and CPOROS() are invoked to calculate water compressibility and pore volume compressibility. For water compressibility, use average value between initial pressure and average reservoir pressure as the working pressure.**

```
              PAvg    = (Pi + PRes) / 2.
              Cw      = Cwater(PAvg, Tem, Salin(J))
              Cp      = Cporos(Poros(J))
```

Step 7: **Constants for dimensionless time *(TimCon)* and dimensionless pseudo-pressure *(PsiCon)* are calculated.**

Note: CMuEff is effective compressibility-viscosity product.

```
              Ct     = (Cw * Swi(J) + Cp)/(1.-Swi(J))
              CMuEff = 2. * (1. - Swi(J)) / (Psii - PsiRes) *
     +                  (Pi/Zi - PRes/Z*(1.-Ct*(Pi-PRes)-WeD(J)))
              TimCon(J) = 0.006328*Perm(J)*365. /
     +                   (Poros(J) * CMuEff * WSpace(J) * 43560.)
              PsiCon(J) = 1422.*(Tem+460.)/(Perm(J)*Thick(J))
```

Step 8: **Pseudo-pressure drops due to previous productions are calculated through numerical convolution (superposition in time).**

Note: Pseudo-pressure drops (*DPsi()*) and loop for convolution are initialized.

```
                 DPsi(J,1) = 0.
                 DPsi(J,2) = 0.
                 DPsi(J,3) = 0.
                 Do I = 1, ITime
```

Note: Time step size (*DT*) is set and dimensionless time (*Tda*) is calculated. Array variable *Time()* stores time data (e.g. 1, 2, 3,...) to be analyzed. The time step size is calculated by subtracting years from two consecutive data points. The size of the first time step is equal to the number of years in the first data point.

```
                    DT = Time(ITime)
                    If (I.gt.1) DT = DT - Time(I-1)
                    Tda = TimCon(J) * DT
```

Note: Loop for well type (*K*: 1=primary, 2=infill once, 3=infill twice) is initialized. Module's number and effective wellbore radius for the corresponding well type are obtained from variables *IMod()* and *Rw()*, respectively.

```
                    Do K = 1, 3
                       Module = IMod(J,K)
                       Rweff  = Rw  (J,K)
```

Note: Recalculate effective wellbore radius (*Rweff*) for fractured reservoirs (modules 2 and 4). For vertical well with infinite conductivity fracture (*Fcd < 0*) or horizontal well (*JTyp=1*) the dimensionless fracture conductivity (*Fcd*) is set to 1E6. Note that the RP Module treats the horizontal well as an infinite conductivity fracture.

```
                          If ((Module.eq.2).or.(Module.eq.4)) then
                             DLen = HalfLn(J,K)
                             SHor = 0.
                             If (JTyp(J,K).eq.0) then
                                Fcd  = Cond(J,K)/(Perm(J)*DLen)
                                If (Fcd .le. 0.) Fcd = 100000.
                                Rweff = DLen
                             Else If (JTyp(J,K) .eq. 1) then
                                DLen  = HorLen(J,K)
                                Fcd   = 100000.
                                Rweff = DLen/2.
                                Ratio = Sqrt(PermV(J)/Perm(J))
                                Rwd  = Rw(J,K)/Thick(J)*Ratio
                                SHor = -2.*Thick(J)/DLen/Ratio *
     +                                 Log(2.*Asin(3.1415926*Rwd))
                             End If
                          End If
```

Note: Sub-program PD() is invoked to calculate dimensionless pressures. Some parameters such as dimensionless area factor (*ARw*), Warren and Root porosity-compressibility ratio (*Omega*), and Warren and Root interporosity flow parameter (*DLam*) are calculated and passed to the PD() sub-program. Note that the Warren and Root parameters (*Omega* and *DLam*) are used to calculate pressure drops in naturally fractured reservoirs (Modules 3 and 4).

```
                        ARw = Sqrt(WSpace(J)*43560.)/Rweff
                        If ((Module.eq.3).or.(Module.eq.4)) then
                            Omega = PorMa(K)/Poros(K)
                            DLam = 12. * PermMa(K)/Perm(K) *
      +                                        (Rweff/FrcSpc(K))**2
                        End If
                        Call PD(Tda,Module,ARw,Fcd,SHor,Omega,DLam,
      +                      ISpeed,Pdw,PdCorn,PdEdge,IErr)
```

Note: The calculated dimensionless pressures are stored in variable *C(p1,p2,p3)*, where *p1* is for pay grade, *p2* is for primary or infill wells, and *p3* is interference term for other wells (1=corner of primary, 2=corner of infill, 3=edge).

```
                        C(J,K,3) = PdEdge
                        C(J,1,2) = PdCorn
                        C(J,2,1) = PdCorn
                        C(J,3,1) = PdEdge
                        C(J,3,2) = PdEdge
                        C(J,K,K) = Pdw
```

Note: Pseudo-pressure drops of the current time step are calculated and added to the one from previous time steps (convolution). The convolution is done only for time steps prior to time step *ITime* (*I<ITime*).

```
                        If (I.lt.ITime) DPsi(J,K) = DPsi(J,K)+PsiCon(J)*
      +                    (DQ(J,1,I)*C(J,K,1)+DQ(J,2,I)*C(J,K,2)+
      +                     DQ(J,3,I)*C(J,K,3)*2.)
```

Note: Loops for well type and time step are closed.

```
                    End Do
                End Do
```

Step 9: **Pay grade loop is closed, program control is returned back to the calling routine (sub-program WDRIVE(), CALCS(), or CNTRL()), and the sub-program CONVLV() is ended.**

```
          End Do
        Return
        End
```

SUB-PROGRAM DIMWE()

LOCATION: MODULE6D.FOR

MAIN THEME: This routine calculates dimensionless cumulative water influx based on tables presented by Van Everdingen and Hurst (1949). The table was adjusted at early time for better accuracy. The calculations are based on the imposed pressure drop from the inner boundary of the aquifer to the initial aquifer pressure. The method assumes that a radial aquifer is external to the gas field. A table look-up method together with linear interpolation technique is utilized.

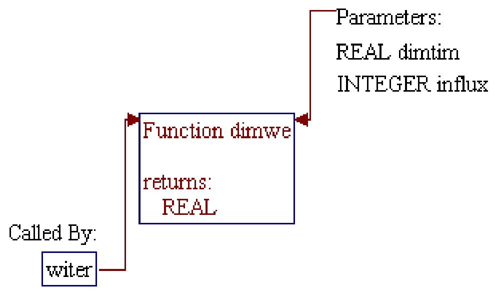
CALLS: None

CALLED BY: WITER() (in file MODULE6C.FOR)
Calculates water influx and based on that decides whether production from a specific water drive reservoir needs to be stopped (if water fills up the reservoir) or not.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variables are declared.**

Note: Name of the sub-program is DIMWE() and the parameters passed to this sub-program are as follows:

Input Parameters:

- *DimTim* Dimensionless time based on the field radius
- *Influx* Constant specifying aquifer size: 0=finite aquifer (Re/Rw=2.5), 1=finite aquifer (Re/Rw=5), 2=infinite aquifer, where Re=external radius (reservoir radius) and Rw=wellbore radius

Output Parameter:

- *DimWe* Dimensionless cumulative water influx

```
      FUNCTION DimWe (DimTim, Influx)
```

Note: Local variables for Van Everdingen and Hurst tables are declared.

```
      Dimension TDw5(39),QD5(39),TDw25(29),QD25(29),
     +          TDInf(100),QDInf(100)
```

Step 2: **Data from Van Everdingen and Hurst tables are assigned to the array variables.**

```
      Data TDw25 /
     +      0.5,      0.6,      0.7,      0.8,       0.9,     1.0,
     +      1.1,      1.2,      1.3,      1.4,       1.5,     1.6,
     +      1.8,      2.0,      2.2,      2.4,       2.6,     2.8,
     +      3.0,      3.4,      3.8,      4.2,       4.6,     5.0,
     +      6.0,      7.0,      8.0,      9.0,      10.0/
      Data QD25 /
     +   1.0244,    1.140,    1.248,    1.348,    1.440,    1.526,
     +    1.605,    1.679,    1.747,    1.811,    1.870,    1.924,
     +    2.022,    2.106,    2.178,    2.241,    2.294,    2.340,
     +    2.380,    2.444,    2.491,    2.525,    2.551,    2.570,
     +    2.599,    2.613,    2.619,    2.622,    2.624/
      Data TDw5 /
     +      2.5,      3.0,      3.5,      4.0,       4.5,      5.0,
     +      5.5,      6.0,      6.5,      7.0,       7.5,      8.0,
     +      8.5,      9.0,      9.5,      10.,       11.,      12.,
     +      13.,      14.,      15.,      16.,       18.,      20.,
     +      22.,      24.,      26.,      28.,       30.,      34.,
     +      38.,      42.,      46.,      50.,       60.,      70.,
     +      80.,      90.,     100./
      Data QD5 /
     +    2.833,    3.195,    3.542,    3.875,     4.193,    4.499,
     +    4.792,    5.074,    5.345,    5.605,     5.854,    6.094,
     +    5.325,    6.547,    6.760,    6.965,     7.350,    7.706,
     +    8.035,    8.339,    8.620,    8.879,     9.338,    9.731,
     +    10.07,    10.35,    10.59,    10.80,     10.98,    11.26,
     +    11.46,    11.61,    11.71,    11.79,     11.91,    11.96,
     +     11.98,    11.99,    12.00/
        Data TDInf /
     +      0.01,    0.015,     0.02,    0.025,      0.03,    0.04,
     +      0.05,     0.06,     0.07,     0.08,      0.09,    0.10,
     +      0.15,      0.2,     0.25,      0.3,       0.4,     0.5,
     +       0.6,      0.7,      0.8,      0.9,        1.,     1.5,
     +        2.,      2.5,       3.,       4.,        5.,      6.,
     +        7.,       8.,       9.,      10.,       15.,     20.,
     +       25.,      30.,      40.,      50.,       60.,     70.,
     +       80.,      90.,     100.,     150.,      200.,    250.,
     +      300.,     400.,     500.,     600.,      700.,    800.,
     +      900.,    1000.,    1500.,    2000.,     2500.,   3000.,
     +     4000.,    5000.,    6000.,    7000.,     8000.,   9000.,
     +    10000.,   15000.,   20000.,   25000.,    30000.,  40000.,
     +    50000.,   60000.,   70000.,   80000.,    90000., 100000.,
     +   150000.,  200000.,  250000.,  300000.,   400000., 500000.,
     +   600000.,  700000.,  800000.,  900000.,  1000000.,1500000.,
     +  2000000., 2500000., 3000000., 4000000.,  5000000.,6000000.,
     +  7000000., 8000000., 9000000.,10000000./
        Data QDInf /
     +  0.117838,   0.1455,   0.1694,   0.1905,   0.2100,   0.2450,
     +    0.2764,   0.3052,   0.3320,   0.3573,   0.3814,   0.4043,
     +    0.5077,   0.5980,   0.6803,   0.7564,   0.8964,   1.0244,
     +    1.1437,   1.2577,   1.3650,   1.4659,   1.5680,   2.0301,
     +    2.4454,   2.8330,    3.200,    3.883,    4.534,    5.148,
     +     5.737,    6.308,    6.861,    7.402,    9.949,    12.32,
     +     14.57,    16.74,    20.88,    24.84,    28.66,    32.37,
     +     35.99,    39.54,    43.03,    59.73,    75.59,    90.87,
     +     105.7,    134.5,    162.2,    189.3,    215.8,    241.8,
     +     267.4,    292.6,    413.6,    531.5,    646.6,    759.0,
     +     975.7,    1188.,    1395.,    1599.,    1800.,    1999.,
     +     2196.,    3146.,    4079.,    4994.,    5891.,    7634.,
     +     9342.,   11030.,   12690.,   14330.,   15950.,   17560.,
     +    25380.,   33080.,   40660.,   48170.,   62670.,   76990.,
     +    91130.,  105100.,  118900.,  132600.,  146200.,  212600.,
     +   278100.,  342700.,  406400.,  531300.,  654400.,  776100.,
     +   896500., 1016000., 1134000., 1252009./
```

Step 3: **Aquifer size is checked. If value of *influx* is out of range, finite aquifer assumption is used (*influx=0*).**

```
        If (Influx.gt.2) Influx = 0
```

Step 4: **Zero value is returned if the specified dimensionless time (*DimTim*) is negative (not specified).**

```
        DimWe = 0.
        If (DimTim.le.0.) Return
```

Step 5: **Early time approximation is utilized if dimensionless time is less than the first entry in *TDInf()* array (*DimTim<TDInf(1)*).**

```
        If (DimTim.le.TDInf(1)) then
                DimWe = Sqrt(4.*DimTim/3.1415926) + DimTim * 0.5
```

Step 6: **If condition in Step 5 is not satisfied, check if conditions for infinite reservoirs are satisfied.**

Note: Late time approximation is utilized if dimensionless time is greater than the last entry in *TDInf()* array (*TDInf(100)*) and one of the following conditions is true:

- *Influx=2* (infinite aquifer)
- *Influx=0* and *DimTim<=TDw25(1)* (finite aquifer and dimensionless time is at most equal to the first entry in *TDw25()* array)
- *Influx=1* and *DimTim<=TDw5(1)* (finite aquifer and dimensionless time is at most equal to the first entry in *TDw25()* array)

```
         Else If ( (Influx.eq.2).or.
      +          ( (DimTim.le.TDw25(1)) .and.(Influx.eq.0) ).or.
      +          ( (DimTim.le.TDw5 (1)) .and.(Influx.eq.1) )) then
                 If (DimTim.gt.TDInf(100)) then
                         DimWe = 2.0180 * DimTim / Log(DimTim)
```

Note: Otherwise, table look-up on infinite reservoir is utilized. The procedure starts with searching the locations of data points for linear interpolation (*I0* and *I1*) using Bi-section technique. Using the data designated by pointers *I0* and *I1*, linear interpolation is performed to calculate dimensionless cumulative water influx (*DimWe*).

```
                 Else
                         I0 = 1
                         I1 = 100
                         DN = 100
                         D  = Log(DN)/Log(2.) + 1.
                         IMax = Int(D)
                         Do I = 1, IMax
                                 If (I1.gt.I0+1) then
                                         I2 = (I0 + I1 ) / 2
                                         If (DimTim.le.TDInf(I2)) then
                                                 I1 = I2
                                         Else
                                                 I0 = I2
                                         End If
                                 End If
                         End Do
                         Y0 = QDInf(I0)
                         Y1 = QDInf(I1)
                         X0 = TDInf(I0)
                         X1 = TDInf(I1)
                         X  = DimTim
                         DimWe = Y0 + (Y1-Y0)/(X1-X0)*(X-X0)
                 End If
```

Step 7: **If conditions in Steps 5 and 6 are not satisfied, check if conditions for finite reservoirs with *Re/Rw=5* are satisfied (*Influx=1*).**

Note: Late time approximation is utilized if dimensionless time is greater than value of *TDw5(39).*

```
        Else If (Influx.eq.1) then
                If (DimTim.gt.TDw5(39)) then
                        DimWe = 12.
```

Note: Otherwise, table look-up on finite reservoir is utilized. The procedure starts with searching the locations of data points for linear interpolation (*I0* and *I1*) using Bi-section technique. The searching procedure is performed only between the first data entry and the 39th data entry of array *TDw5().* Using the data designated by pointers *I0* and *I1*, linear interpolation is performed to calculate dimensionless cumulative water influx (*DimWe*).

```
                Else
                        I0 = 1
                        I1 = 39
                        DN = 39
                        D  = Log(DN)/Log(2.) + 1.
                        IMax = Int(D)
                        Do I = 1, IMax
                                If (I1.gt.I0+1) then
                                        I2 = (I0 + I1 ) / 2
                                        If (DimTim.le.TDw5(I2)) then
                                                I1 = I2
                                        Else
                                                I0 = I2
                                        End If
                                End If
                        End Do
                        Y0 = QD5 (I0)
                        Y1 = QD5 (I1)
                        X0 = TDw5(I0)
                        X1 = TDw5(I1)
                        X  = DimTim
                        DimWe = Y0 + (Y1-Y0)/(X1-X0)*(X-X0)
                End If
```

Step 8: **If conditions in Steps 5, 6, and 7 are not satisfied, the reservoir type will be finite with aquifer size of *Re/Rw=2.5*.**

Note: Late time approximation is utilized if dimensionless time is greater than value of *TDw25(29).*

```
        Else
                If (DimTim.gt.TDw25(29)) then
                        DimWe = 2.625
```

Note: Otherwise, table look-up on finite reservoir is utilized. The procedure starts with searching the locations of data points for linear interpolation (*I0* and *I1*) using Bi-section technique. The searching procedure is performed only between the first data entry and the 29th data entry of array *TDw25().* Using the data designated by pointers *I0* and *I1*, linear interpolation is performed to calculate dimensionless cumulative water influx (*DimWe*).

```
            Else
                    I0 = 1
                    I1 = 29
                    DN = 29
                    D  = Log(DN)/Log(2.) + 1.
                    IMax = Int(D)
                    Do I = 1, IMax
                            If (I1.gt.I0+1) then
                                    I2 = (I0 + I1 ) / 2
                                    If (DimTim.le.TDw25(I2)) then
                                            I1 = I2
                                    Else
                                            I0 = I2
                                    End If
                            End If
                    End Do
                    Y0 = QD25 (I0)
                    Y1 = QD25 (I1)
                    X0 = TDw25(I0)
                    X1 = TDw25(I1)
                    X  = DimTim
                    DimWe = Y0 + (Y1-Y0)/(X1-X0)*(X-X0)
            End If
    End If
```

Step 9: **Program control is returned back to the calling routine (sub-program WITER()) and the sub-program DIMWE() is ended.**

```
    Return
    End
```

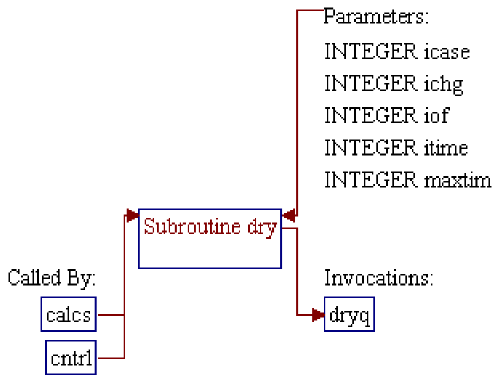
SUB-PROGRAM DRY()

LOCATION:	MODULE6A.FOR
MAIN THEME:	This routine is a type curve module for dry coal and dry shale reservoirs that drives sub-routine DRYQ() to calculate gas flow rates. The module implements material balance directly to compute average reservoir pressure in each pay grade.
CALLS:	DRYQ() (in file MODULE6A.FOR) Calculates gas flow rates for dry coal and dry shale reservoirs based on bottom hole pressure.

CALLED BY:	CALCS() (in file MODULE6D.FOR)
	Performs numerical convolution and solves for pressure and flow rates at each time step to generate type curves.
	
	CNTRL() (in file MODULE6D.FOR)
	Initializes minimum pressures, maximum rates, skins, infill wells ON/OFF, etc.

READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included.**

Note: Name of the sub-program is DRY() and the parameters passed to this sub-program are as follows:

•	*ITime*	Time step number
•	*ICase*	Case number: 1=primary, 2=automatic refrac, 3=automatic infill one time.
•	*IChg*	Flag to indicate whether development type change has taken place: 0=primary wells only, 1=one set of infills, 2=two sets of infills (not currently used), 3= primary wells after restimulation
•	*MaxTim*	Maximum number of time steps
•	*IOF*	Flag to indicate type of calculation: 0=normal, 1=open flow calculation

```
      SUBROUTINE DRY (ITime,ICase,IChg,MaxTim,IOF)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'type2.h'
      include 'type5.h'
      include 'type3.h'
      include 'type4.h'
      include 'type8.h'
      include 'type7.h'
      include 'type6.h'
      include 'type10.h'
```

Step 2: **Amount of gas adsorbed at current time level is calculated.**

Note: Pay grade loop and variables are initialized. Bulk volume (*VB*) is calculated.

```
      Do I = 1, 3
          Sum = 0.
          B0  = 0.
          F0  = 0.
          VB  = WSpace(I) * Thick(I)
```

Note: Time step loop, previous time level (*T*), initial pressure (*Pi*), and average reservoir pressure (*P*) are initialized.

```
             Do J = 1, ITime
                 T = 0.
                 If (J.gt.1) T = Time (J-1)
                 Pi = Pinit(I)
                 P  = PreAvg(I,J)
```

Note: Equilibrium amount of gas adsorbed (*F*), steady-state sorption rate (*B*), time step to sorption time constant ratio (*DT*), and factor *E* (exponential of *DT*) at time step *J* are calculated. This module implements dual-porosity approach (non-equilibrium sorption approach) by taking into consideration sorption time constant (*TDes()*).

```
                 F  = VL(I) * (Pi/(Pi+PL(I)) - P/(P+PL(I)))
                 B  = (F-F0)/(Time(J)-T)
                 DT = (Time(ITime)-T)/(TDes(I)/365.)
                 E  = 0.
                 If (DT.lt.30.) E = exp(-DT)
```

Note: Amount of gas adsorbed (*ADesrb()*) is calculated using non-equilibrium formulation.

```
                 If (J.ne.ITime) then
                     Sum = Sum + E * (B-B0)
                 Else
                     ADesrb(I) =(Sum+F0/DT*(1.-E)-E*B0)*VB*RhoMa(I)/1000.
                     BDesrb(I) = (1. - (1.-E)/DT) * VB * RhoMa(I) / 1000.
                     ADesrb(I) = ADesrb(I)+BDesrb(I)*VL(I)*Pi/(Pi+PL(I))
                 End If
```

Note: Values of *B* and *F* are stored for calculation at the next time step. The time step and pay grade loops are closed.

```
                 B0 = B
                 F0 = F
             End Do
         End Do
```

Step 3: **Sub-program DRYQ() is invoked to calculate gas flow rate at minimum allowable wellhead pressure. The well is shut in if the resulting flow rate is too low.**

Note: Using a minimum wellhead pressure *(PreMin)* with normal calculation (*IOF=0*, not an open flow calculation), gas flow rate is calculated. If the flow rate is less than 1 MCFD, the well is shut in by setting *KShut()* equals to 1.

```
         Pwh = PreMin
```

```
IOF = 0
Call DryQ (Pwh, QTotal, ITime, IChg, IOF)
Do J = 1, 3
       If (Qg(J,1,ITime).lt.1.) KShut(J) = 1
End Do
```

Step 4: **Gas flow rate is recalculated if restimulation (refracturing) or infilling is required.**

Note: If the total gas flow rate (*QTotal*) obtained from Step 3 is less than user specified maximum gas rate (*RatMax*) and the development case has not yet changed (*IChg=0*), sub-program DRYQ() is invoked to recalculate gas rate with refrac (*ICase=2*) or infills *(ICase=3)* option.

```
If ((QTotal.lt.RatMax).and.(IChg.eq.0)) then
       If (ICase.eq.2) then
              IChg = 3
              Call DryQ (Pwh, QTotal, ITime, IChg, IOF)
       Else If (ICase.eq.3) then
              IChg = 1
              Call DryQ (Pwh, QTotal, ITime, IChg, IOF)
       End If
End If
```

Step 5: **Gas flow rate is recalculated iteratively if total gas flow rate based on minimum allowable wellhead pressure (*QTotal*) is greater than user specified maximum gas rate (*RatMax*).**

Note: Perform 5 Bi-section iterations to get good estimate of wellhead pressure for Newton-Raphson iteration. For the Bi-section iteration, the current wellhead pressure (*Pwh*) is used as minimum pressure (*P0*). The maximum pressure (*P1*) is set to the highest initial pressure from the three pay grades (*Pinit()*) for the first time step or is set to the highest average reservoir pressure from the three pay grades (*PreAvg()*) of previous time step.

```
             P0 = Pwh
             P1 = Max(Pinit(1),Pinit(2),Pinit(3))
             If (ITime.gt.1) P1 = Max(PreAvg(1,ITime-1),
     +                     PreAvg(2,ITime-1),PreAvg(3,ITime-1))
             Do I = 1, 5
                Pwh = (P1+P0) / 2.
                Call DryQ (Pwh, QTotal, ITime, IChg, IOF)

                If (QTotal.gt.RatMax) then
                    P0 = Pwh
                Else
                    P1 = Pwh
                End If
             End Do
```

Note: Using the middle point of pressures *P0* and *P1* (from the Bi-section iteration) as the wellhead pressure, the iterative procedure for flow rate calculation is continued using a maximum of 10 Newton-Raphson iterations. A central difference approach is used in numerical derivative with epsilon pressure (*DP*) of 5 psi.

```
              Pwh = (P1+P0) / 2.
              JSolv = 0
              DP = 5.
              Do I = 1, 10
                If (JSolv.eq.0) then
                  Pwh = Pwh + DP
                  Call DryQ (Pwh, QTotal, ITime, IChg, IOF)
                  F1 = QTotal - RatMax
                  Pwh = Pwh - DP
                  Call DryQ (Pwh, QTotal, ITime, IChg, IOF)
                  F  = QTotal - RatMax
                  If (abs(F).le.0.1) then
                      JSolv = 1
                  Else
                      If (abs(F1-F) .lt. 0.000001) then
                           DP = DP * 2.
                      Else
                            Pwh = Pwh - F * DP / (F1 - F)
                      End If
                  End If

                End If
              End Do
      End If
```

Step 6: **Sub-program DRYQ() is invoked one more time to determine gas flow rate based on open flow calculation (*IOF=1*).**

```
      IOF = 1
      Call DryQ (Pwh, QTotal, ITime, IChg, IOF)
```

Step 7: **The program control is returned back to the calling routine (sub-program CALCS() or CNTRL()) and the sub-program DRY() is ended.**

```
      Return
      End
```

SUB-PROGRAM DRYQ()

LOCATION: MODULE6A.FOR

MAIN THEME: This routine calculates gas flow rates for dry coal and dry shale reservoirs. The routine solves for flow rates of the primary well case (no infilling) under pressure constraint. The minimum flow rate constraint is zero and the maximum is based on the sandface pressure from previous withdrawals.

CALLS: PSI() (in file MODULE6B.FOR)
Performs table look-up of real gas potential (psudo-pressure) given pressure.

PWELL() (in file MODULE6B.FOR)
Calculates bottomhole pressure, wellhead pressure, or flow rate based on the difference between well head pressure and bottom hole pressure of the well using Smith's formula.

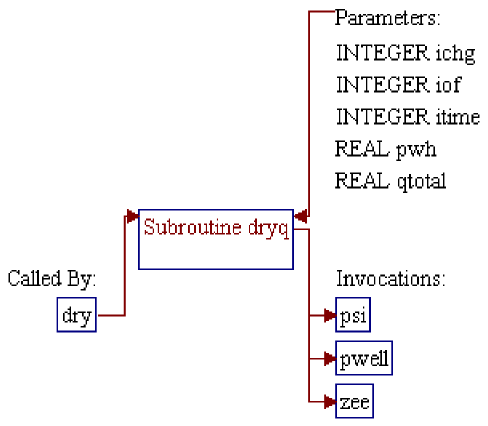
ZEE() (in file MODULE6C.FOR)
Performs table look-up linear interpolation of Z-factor as a function of pressure.

CALLED BY: DRY() (in file MODULE6A.FOR)
Computes performances of dry coal and shale reservoirs using material balance approach.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included.**

Note: Name of the sub-program is DRYQ() and the parameters passed to this sub-program are as follows:

•	*Pwh*	Wellhead pressure (psia)
•	*QTotal*	Total gas production from field (MCF/D)
•	*ITime*	Time step number
•	*IChg*	Flag to indicate whether development type change has taken place: 0=primary wells only, 1=one set of infills, 2=two sets of infills (not currently used), 3= primary wells after restimulation
•	*IOF*	Flag to indicate type of calculation: 0=normal, 1=open flow calculation

```
      SUBROUTINE DRYQ (Pwh, QTotal, ITime, IChg, IOF)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
```

Step 2: **Time step size *DT* is set.**

Note: Array variable *Time()* stores time data (e.g. 1, 2, 3,...) to be analyzed. The time step size is calculated by subtracting years from two consecutive data points. The size of the first time step is equal to the number of years in the first data point.

```
      DT = Time(ITime)
      If (ITime.gt.1) DT = DT - Time(ITime-1)
```

Step 3: **Loop for pay grades (loop *J*) and flow rates are initialized.**

Note: Gas production rate use three-parameter array variable *Qg(p1,p2,p3)* where *p1* is for pay grades (1,2,3), *p2* is for primary or infill (1=primary, 2=infill), and *p3* is for time step (1,2,3,...). Initialization of this variable is controlled by flag *Kshut()* (0=well is producing, >0=well is shut in). Note that variable *Qw()* is for water production rate (not used in this routine) and it is set to zero.

```
         Do J = 1, 3
            If (KShut(J).gt.0) then
                Qg(J,1,ITime) = 0.
                Qw(J,1,ITime) = 0.
                Qg(J,2,ITime) = 0.
                Qw(J,2,ITime) = 0.
            Else
```

Step 4: **Information on skin factor (*S*), effective wellbore radius (*Rweff*), and the horizontal well length or fracture half-length (*DLen*) are obtained from input data variables.**

Note: The skin factor is obtained from three-parameter array variable *Skin(p1,p2,p3)* where:

•	*p1*	Parameter for pay grade (1,2,3)
•	*p2*	Parameter for primary or infill wells (1=primary, 2=infill)
•	*p3*	Parameter for stimulation type (1=no refrac, 2=with refrac)

Note that parameters *p2* and *p3* are dependent on each other because the current version of the RP Module does not consider infilling with refracturing as a development case. Therefore, for *p2=2*, the *p3* can only have a value of *1*. The effective well bore radius and the horizontal well length/fracture half-length are obtained from variables *Rw(), HalfLn()*, respectively.

```
                S = Skin(J,1,1)
                If (IChg.eq.3) S = Skin(J,1,2)
                Rweff  = Rw  (J,1)
                DLen   = HalfLn(J,1)
```

Step 5: **Effective wellbore radius for fractured or horizontal well is recalculated.**

Note: The value of *DLen* of greater than *1* indicates either the system is fractured or the well is horizontal. In this case, the dimensionless fracture conductivity (*Fcd*) is calculated, and if the value is negative (for infinite conductivity fracture) the *Fcd* is set to 100000.

```
            If (DLen.gt.1.) then
                Fcd   = Cond(J,1) / (Perm(J)*DLen)
                If (Fcd .le. 0.) Fcd = 100000.
```

Note: The following codes calculate the effective wellbore radius for two different wells: fractured vertical well (*JTyp()=0*), and horizontal well without fracture (*JTyp()=1*).

```
                SHor  = 0.
                Rweff = DLen/2.
                If (JTyp(J,1) .eq. 1) then
                    DLen  = HorLen(J,1)
                    Ratio = Sqrt(PermV(J)/Perm(J))
                    Rwd   = Rw(J,1)/Thick(J)*Ratio
                    SHor  = -2.*Thick(J)/DLen/Ratio*
     +                       Log(2.*Asin(3.1415926*Rwd))
                    Rweff = DLen / 4. * Exp (-SHor)
                Else
                    Rweff = Rweff / (1. + 1.71 / Fcd)
                End If
                If (Rweff .lt. Rw(J,1)) Rweff = Rw(J,1)
            End If
```

Step 6: **Productivity index (*Qcon*) is calculated.**

Note: The productivity index is calculated using a standard well bore equation. For infill wells (*IChg=1*), drainage area (*Re*) is assumed to be equally split with equal skin factors and the productivity index is multiplied by *2* (for two wells).

```
            Re   = Sqrt(43560.*WSpace(J)/3.1415926)
            PCon = Max(Log(Re/Rweff) - 0.75 + S, 1.)
            QCon = Perm(J)*Thick(J) / (1422.*(Tem+460.)*PCon)
            If (IChg.eq.1) then
                Re   = Re / Sqrt(2.)
                PCon = Max(Log(Re/Rweff) - 0.75 + S, 1.)
                QCon = 2.*Perm(J)*Thick(J)/(1422.*(Tem+460.)*PCon)
            End If
```

Step 7: **Cumulative gas production is calculated.**

Note: Sub-program ZEE() (located in fie MODULE6C.FOR) is invoked to determine gas Z-factor at initial pressure (*Pinit()*).

```
            Pi   = Pinit(J)
            Zi   = Zee(Pi,NArray,PreAry,ZAry)
```

Note: Pore volume compressibility (*Cp*) for coal and shale reservoirs is assumed to be 0.00005/psi. This variable will be utilized later in material balance calculations. Depth of the reservoir (*Depth1()*) is assigned to working variable *Dep* to be used later in sub-program PWELL(). Volume of gas at current time (*G1i*) is calculated.

```
              Cp     = .00005
              Dep  = Depth1(J)
              G1i  = 43560. * WSpace(J) * Thick(J) * Poros(J) *
     +              (1.-Swi(J)) * 520./(Tem+460.)/ 14.7 /1000.*Pi/Zi
```

Note: Cumulative gas production up to the last time step (*Gp0*) is calculated. This calculation utilizes three-parameter arrays of cumulative gas production *CumGas(p1,p2,p3)* where:

•	*p1*	Parameter for pay grade (1,2,3)
•	*p2*	Parameter for primary or infill wells (1=primary, 2=infill once, 3=infill twice)
•	*p3*	Parameter for time steps (1,2,...)

```
              Gp0  = 0.
              If (ITime.gt.1) Gp0 = CumGas(J,1,ITime-1) +
     +                CumGas(J,2,ITime-1) + 2.*CumGas(J,3,ITime-1)
```

Step 8: **Data for the first part of iterative procedure (Bi-section method) to calculate average reservoir pressure (*P*) are prepared.**

Note: Minimum pressure (*P0*) for Bi-section iteration is set to 14.7 psia if the calculation is for open flow (*IOF=1*). Otherwise, sub-program PWELL() is invoked with zero flow rate (*QMin=0*) to calculate bottom hole pressure as the minimum pressure (*P0*). In the later case, the sub-program PWELL() is asked to calculate bottom hole pressure by setting the sixth parameter of the sub-program equals to “1”. The reservoir type (*KTyp*) is obtained from variable *KunCon()* which can have a value of:

•	*0*	Dry coal reservoir
•	*1*	Wet coal reservoir
•	*2*	Dry shale reservoir
•	*3*	Wet shale reservoir

```
              QMin = 0.
              KTyp = KUnCon(J)
              If (IOF .eq. 0) then
                 Call PWELL(Pwh,P0,QMin,Deriv,Dep,1,IErr,KTyp,J)
```

```
                  Else
                      P0 = 14.7
                  End If
```

Note: Maximum pressure for Bi-section iteration (*P1*) is obtained from the previous average reservoir pressure (*PreAvg(J,Itime-1)*). Average pressure for the current time step will be between the minimum and maximum pressures (*P0 < P < P1*).

```
                  P1   = Pinit(J)
                  If (ITime .gt. 1) P1 = PreAvg(J,ITime-1)
```

Step 9: **The first part of iterative procedure to calculate average reservoir pressure (*P*) is performed using 5 Bi-section iterations.**

Note: Iteration loop (*I*) is initialized. The average reservoir pressure (*P*) at current Bi-section iteration level is taken as the middle point between the minimum and maximum pressures.

```
                  Do I = 1, 5
                      P  = (P0 + P1) / 2.
```

Note: Flow rate at current time step based on gas material balance equation (*Q1* or *Q*) is calculated. First the cumulative gas production at current time step (*Gp*) is calculated. Cumulative gas sorption for coal reservoir is added to the cumulative gas production equation by implementing Langmuir equation (*ADesrb(J) - BDesrb(J) * VL(J) * P/(P+PL(J))*). The gas flow rate is then calculated based on the current and previous time-level cumulative gas productions. If the change of development case has taken place (from primary to infill once, *IChg=1*), the well flow rate is divided by 2.

```
                      Z  = Zee(P,NArray,PreAry,ZAry)
                      Gp = G1i * (1.-(1.-Cp*(Pi-P))*P/Z/(Pi/Zi)) +
     +                     ADesrb(J) - BDesrb(J) * VL(J) * P/(P+PL(J))
                      Q1 = (Gp - Gp0) / (365.*DT)
                      Q  = Q1
                      If (IChg.eq.1) Q = Q1 / 2.
```

Note: Sub-program PSI() (located in file MODULE6B.FOR) is invoked twice to determine the average pseudo-pressure (*PsiAvg*) as a function of average pressure (*P*) and bottom-hole pseudo-pressure (*PsiBh*) as a function of bottom hole pressure (*Pbha*). Flow rate based on the well bore equation (*Q2*) is then calculated.

```
                     PsiAvg = Psi(P,NArray,PreAry,PsiAry)
                     PsiBh  = Psi(Pbha,NArray,PreAry,PsiAry)
                     Q2     = QCon * (PsiAvg - PsiBh)
```

Note: The minimum or maximum pressures (*P0* or *P1*) is updated based on the difference between *Q1* and *Q2* and the Bi-section iterative process is repeated until the iteration counter (*I*) equals to 5. At the end of this iteration, the *P*, *P0*, and *P1* are expected to be relatively close to each other so that the iteration can be continued using the Newton-Raphson iterative procedure which requires a good initial estimation of *P*.

```
                     If (Q1.gt.Q2) then
                         P0 = P
                     Else
                         P1 = P
                     End If
                 End Do
```

Step 10: **The second part of the iterative procedure to calculate average reservoir pressure (*P*) is performed using Newton-Raphson procedure up to 10 iterations.**

Note: The Newton-Raphson iterative procedure is initialized. Initial guess of pressure (*P*) is taken as the middle point between pressures *P0* and *P1* from the Bi-section iteration. Epsilon pressure for numerical derivative (*DP*) is set equal to *2* psi and convergence flag (*JSolv*) is set to zero to indicate non-convergence condition.

```
                 P  = (P0 + P1) / 2.
                 DP = 2.
                 JSolv = 0
```

Note: Iteration loop (*I*) is initialized. The iteration process is repeated if the results do not meet the convergence criterion.

```
                 Do I = 1, 10
                   If (JSolv .lt. 2) then
```

Note: The following codes calculate flow rate based on material balance (*Q1*) and flow rate based on well bore equation (*Q2*) using the same procedure as in the Bi-section iteration evaluated at reservoir pressure of *P*. The difference between *Q1* and *Q2* is stored at variable *F*.

```
                     Z  = Zee(P,NArray,PreAry,ZAry)
                     Gp = G1i * (1.-(1.-Cp*(Pi-P))*P/Z/(Pi/Zi)) +
     +                    ADesrb(J) - BDesrb(J) * VL(J) * P/(P+PL(J))
                     Q1 = (Gp - Gp0) / (365.*DT)
                     Q = Q1
                     If (IChg.eq.1) Q = Q1 / 2.
                     If (IOF .eq. 0) then
                         Call PWELL(Pwh,Pbh,Q,Deriv,Dep,1,IErr,KTyp,J)
                     Else
                         Pbh = 14.7
                     End If
                     PsiAvg = Psi(P,NArray,PreAry,PsiAry)
                     PsiBh  = Psi(Pbh,NArray,PreAry,PsiAry)
                     Q2     = QCon * (PsiAvg - PsiBh)
                     F      = Q1 - Q2
```

Note: The convergence is checked by evaluating the value of dependent variable (*F*). If the convergence is achieved (value of *F* is less than 0.1 MCF/well), the convergence flag *JSolv* is set to 1 (intermediate condition) and it will change to 2 in the next iteration to indicate the termination of the iteration. If the convergence is not yet achieved, the process is continued to calculate *Q1*, and *Q2* evaluated at pressure *P+DP*, and the difference between *Q1* and *Q2* is stored at variable *FP*.

```
                     If (JSolv.eq.1) then
                         JSolv = 2
                     Else If (abs(F) .lt. 0.1) then
                         JSolv = 1
                     Else
                         P  = P + DP
                         Z  = Zee(P,NArray,PreAry,ZAry)
                         Gp = G1i * (1.-(1.-Cp*(Pi-P))*P/Z/(Pi/Zi)) +
     +                    ADesrb(J) - BDesrb(J) * VL(J) * P/(P+PL(J))
                         Q1 = (Gp - Gp0) / (365.*DT)
                         Q = Q1
                         If (IChg.eq.1) Q = Q1 / 2.
                         If (IOF .eq. 0) then
                           Call PWELL(Pwh,Pbh,Q,Deriv,Dep,1,IErr,KTyp,J)
                         Else
                           Pbh = 14.7
                         End If
                         PsiAvg = Psi(P,NArray,PreAry,PsiAry)
                         PsiBh  = Psi(Pbh,NArray,PreAry,PsiAry)
                         Q2     = QCon * (PsiAvg - PsiBh)
                         FP     = (Q1 - Q2) – F
```

Note: The reservoir pressure is set back to its original value, pressure improvement (*DelP*) is calculated, the pressure (*P*) is updated, and the iteration process is continued.

```
                         P = P – DP
                         Del P = - F * DP / FP
                         PT = P + DelP
                         If (PT.lt.P0) then
                             P = P + (P0-P) * 0.75
                         Else If (PT.gt.P1) then
                             P = P + (P1-P) * 0.75
```

```
                        Else
                            P = PT
                        End If
                    End If
                  End If
                End Do
```

Note: If the convergence is still not yet achieved (*JSolv=0*), the iteration process is continued with 15 more Bi-section iterations.

```
             If (JSolv.eq.0) then
                Do I = 1, 15
                    P  = (P0 + P1) / 2.
                    Z  = Zee(P,NArray,PreAry,ZAry)
                    Gp = G1i * (1.-(1.-Cp*(Pi-P))*P/Z/(Pi/Zi)) +
     +                   ADesrb(J) - BDesrb(J) * VL(J) * P/(P+PL(J))
                    Q1 = (Gp - Gp0) / (365.*DT)
                    Q = Q1
                    If (IChg.eq.1) Q = Q1 / 2.
                    If (IOF .eq. 0) then
                        Call PWELL(Pwh,Pbh,Q,Deriv,Dep,1,IErr,KTyp,J)
                    Else
                        Pbh = 14.7
                    End If
                    PsiAvg = Psi(P,NArray,PreAry,PsiAry)
                    PsiBh  = Psi(Pbh,NArray,PreAry,PsiAry)
                    Q2     = QCon * (PsiAvg - PsiBh)
                    If (Q1.gt.Q2) then
                        P0 = P
                    Else
                        P1 = P
                    End If
                End Do
             End If
```

Step 11: **Average reservoir pressure, bottom hole pressure, wellhead pressure, gas flow rate, and cumulative gas production for the case without open flow calculation (*IOF=0*) are obtained or calculated and stored to type curve variables.**

Note: Get type of unconventional reservoir (*KunCon()*) and store the value to working variable *KTyp*. In this routine, the *KTyp* is used to indicate dry coal or dry shale reservoir. *KunCon()* can have a value of:

•	*0*	Dry coal reservoir
•	*1*	Wet coal reservoir
•	*2*	Dry shale reservoir
•	*3*	Wet shale reservoir

```
                KTyp = KUnCon(J)
```

Note: The following codes store pressure and flow rates results to the type curve variables. Bottom hole pressure (*Prbh()*), wellhead pressure (*Prwh()*) and gas flow rate (*Qg*) for the case with infill wells (*IChg=1*) are updated accordingly.

```
                 If (IOF .eq. 0) then
                     PreAvg(J,ITime) = P
                     Qg(J,1,ITime)   = Q
                     Prwh(J,1,ITime) = Pwh
                     Prbh(J,1,ITime) = Pbh
                     Prbh(J,2,ITime) = Pbh + (1.+0.25/PCon)*(P-Pbh)
                     Prbh(J,3,ITime) = P
                     If (IChg.eq.1) then
                         Qg(J,2,ITime)   = Q
                         Prwh(J,2,ITime) = Pwh
                         Prbh(J,2,ITime) = Pbh
                         Prbh(J,3,ITime) = P
                     End If
```

Note: Sub-program PWELL() (located in file MODULE6B.FOR) is invoked to calculate wellhead pressure. An integer “2” in the sixth parameter of the sub-program PWELL() tells the routine to calculate wellhead pressure given the bottom hole pressure.

```
                     Do K = 1, 3
                         Qk = Qg(J,K,ITime)
                         Pk = Prbh(J,K,ITime)
                         Call PWELL(Pwk,Pk,Qk,Deriv,Dep,2,
     +                              IErr,KTyp,J)
                         Prwh(J,K,ITime) = Pwk
                     End Do
```

Note: Cumulative gas production is calculated.

```
                     If (ITime.eq.1) then
                         CumGas(J,1,ITime) = Qg(J,1,ITime) * DT * 365.
                         CumGas(J,2,ITime) = Qg(J,2,ITime) * DT * 365.
                     Else
                         CumGas(J,1,ITime) = CumGas(J,1,ITime-1)+
     +                                       Qg(J,1,ITime) * DT * 365.
                         CumGas(J,2,ITime) = CumGas(J,2,ITime-1)+
     +                                       Qg(J,2,ITime) * DT * 365.
                     End If
```

Step 12: **For the case with open flow calculation, the gas flow rate is stored to absolute open flow variable (*CAOF()*). The**

Note: After storing the value of gas flow rate to the CAOF(), the pay grade loop (*J*) is closed.

```
                 Else
                     CAOF(J,1,ITime) = Q
                     If (IChg.eq.1) CAOF(J,2,ITime) = Q
                 End If
```

```
             End If
          End Do
```

Step 13: **Finally, total gas flow rate (*QTotal*) is calculated for each pay grade, the program control is returned back to the calling routine (sub-program DRY()), and the sub-program DRYQ() is ended.**

```
          QTotal = 0.
          Do I=1,3
                  QTotal = QTotal + Area(I) / WSpace(I) *
       +              (Qg(I,1,ITime) + Qg(I,2,ITime) + Qg(I,3,ITime)*2.)
          End Do
          Return
          End
```

SUB-PROGRAM ERRFN()

LOCATION: MODULE6D.FOR

MAIN THEME: This routine calculates error function based on a polynomial approximation from Abramowitz, M. and Stegun, I.A., Handbook of Mathematical Function with Formulas, Graphs and Mathematical Tables, National Bureau of Standards Applied Mathematics Series 55, June, 1964 (10th Printing Dec., 1972, with Corrections).

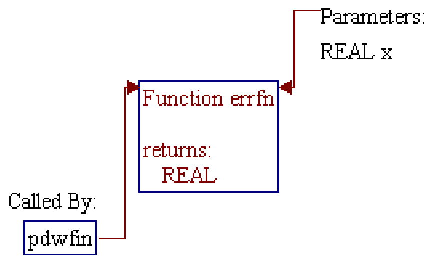
CALLS: None

CALLED BY: PDWFIN() (in file MODULE6B.FOR)
Calculates dimensionless pressure for a well with a finite conductivity fracture producing at a constant rate in an infinite reservoir.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variables are declared.**

Note: Name of the sub-program is ERRFN() and the parameters passed to this sub-program are as follows:

Input Parameter:

•	*x*	Argument to the error function

Output Parameter:

•	*ErrFn*	Calculated error function

```
      FUNCTION ErrFn (x)
```

Note: Local variables are declared.

```
      Double Precision a1, a2, a3, a4, a5, p, t, f
```

Step 2: **Constants for the polynomial equation are assigned.**

```
      Data a1,a2,a3,a4,a5,p/
   +    0.254829592d0, -0.284496736d0, 1.421413741d0, -1.453152027d0,
   +    1.061405429d0,  0.3275911d0/
```

Step 3: **Absolute value of the argument is stored in variable *z*.**

```
      z = Abs(x)
```

Step 4: **Check if the argument is outside the range *–5<x<5*. If *x* is outside the allowable range, return +1 or –1 as the error function.**

```
      If (z .gt. 5.) then
              ErrFn = 1.
```

Step 5: **Taylor series expansion is utilized if the argument is within the range *–0.1<x<01*.**

```
      Else If (z .lt. 0.1) then
              ErrFn = 1.1283792 * (z - z**3/3. + z**5/10.)
```

Step 6: **Otherwise, Abramowitz and Stegun approximation is used.**

```
      Else
              t = 1.d0 / (1.d0 + p * z)
              f = 1.d0 - t * (a1 + t * (a2 + t * (a3 + t * (a4 +
     +                t * a5)))) * Exp (-z**2)
              ErrFn = Real(f)
      End If
```

Step 7: **The sign of the error function is corrected if the argument is negative.**

```
      If (x .lt. 0.) ErrFn = -ErrFn
```

Step 8: **Program control is returned back to the calling routine (sub-program PDWFIN()) and the sub-program ERRFN() is ended.**

```
      Return
      End
```

SUB-PROGRAM EXPINT()

LOCATION: MODULE6D.FOR

MAIN THEME: This routine calculates exponential integral based on a polynomial approximations from Abramowitz, M. and Stegun, I.A., Handbook of Mathematical Function with Formulas, Graphs and Mathematical Tables, National Bureau of Standards Applied Mathematics Series 55, June, 1964 (10th Printing Dec., 1972, with Corrections).

CALLS: None

CALLED BY: PD() (in file MODULE6B.FOR)
Calculates dimensionless pressure functions for different reservoir systems.

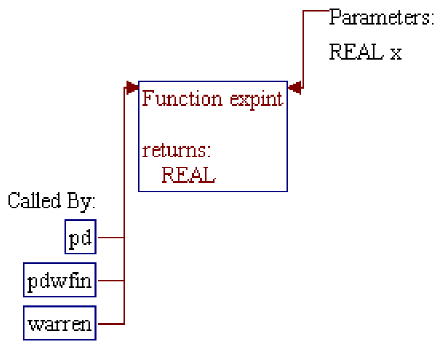
PDWFIN() (in file MODULE6B.FOR)
Calculates dimensionless pressure for a well with a finite conductivity fracture producing at a constant rate in an infinite reservoir.

WARREN() (in file MODULE6C.FOR)
Calculates the difference in dimensionless pressures between a conventional reservoir and a naturally fractured reservoir using Warren and Root approach.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variables are declared.**

Note: Name of the sub-program is EXPINT() and the parameters passed to this sub-program are as follows:

Input Parameters:

- *x* Argument to the exponential integral

Output Parameter:

- *ExpInt* Calculated exponential integral

```
      FUNCTION   ExpInt (x)
```

Note: Local variables are declared.

```
      Double Precision a0,a1,a2,a3,a4,a5,b1,b2,b3,b4,c1,c2,c3,c4,z
```

Step 2: **Constants for the polynomial equations are assigned.**

```
      Data a0,a1,a2,a3,a4,a5,b1,b2,b3,b4,c1,c2,c3,c4/
     +    -0.57721566d0, 0.99999193d0,  -0.24991055d0,   0.05519968d0,
     +    -0.976004d-2,  0.107857d-2,
     +    8.5733287401d0,1.8059016973d1, 8.6347608925d0, 0.2677737343d0,
     +    9.5733223454d0,2.56329561486d1,2.10996530827d1,3.9584969228d0/
```

Step 3: **Check if the argument is outside the range *0<x<90.1*. If *x* is outside the allowable range, return 0 as the exponential integral.**

```
      If ((x.le.0.) .or. (x.ge.90.1)) then
              ExpInt = 0.
              Return
      End If
```

Step 4: **Abramowitz and Stegun approximations are utilized to calculate exponential integral.**

```
      z = Dble(x)
      If (x .le. 1.) then
              ExpInt = (a0+z*(a1+z*(a2+z*(a3+z*(a4+z*a5)))) - Log(z))
      Else
              ExpInt = ((((( z + b1) * z + b2) * z + b3) * z + b4)
     +                  /(((( z + c1) * z + c2) * z + c3) * z + c4)
     +                  / z * Exp(-z))
      End If
```

Step 5: **Program control is returned back to the calling routine (sub-program PD(), PDWFIN(), or WARREN()) and the sub-program EXPINT() is ended.**

```
        Return
        End
```

SUB-PROGRAM FRICTN()

LOCATION: MODULE6D.FOR

MAIN THEME: This routine calculates Moody friction factor using Colebrook-White correlation. Newton-Raphson procedure is utilized to solve the non-linear equation.

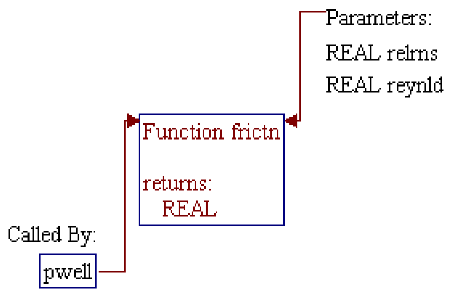
CALLS: None

CALLED BY: PWELL() (in file MODULE6B.FOR)
Calculates bottomhole pressure, wellhead pressure, or flow rate based on the difference between well head pressure and bottom hole pressure of the well using Smith's formula.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared.**

Note: Name of the sub-program is FRICTN() and the parameters passed to this sub-program are as follows:

Input Parameters:

• *Reynld*	Reynold number
• *RelRns*	Relative roughness

Output Parameter:

• *Frictn*	Moody friction factor

```
FUNCTION Frictn (Reynld, RelRns)
```

Step 2: **Friction factor of 1 is returned if Reynold number is less than 64.**

```
If (Reynld .lt. 64.) then
        Frictn = 1.
        Return
End If
```

Step 3: **Laminar flow formula is used if Reynold number is less than 2500.**

```
If (Reynld .lt. 2500.) then
        Frictn = 64./Reynld
        Return
End If
```

Step 4: **Colebrook-White correlation is utilized if Reynold number is higher than 2500.**

Note: 3 Newton-Raphson iterations on *x=1/F**0.5*, starting at initial guess of *F=1/36*. Fx is the objective function, and *Deriv* is the derivative of the function with respect to *x*.

```
x=6
Do I=1,3
        Fx = x + 0.868589 * Log (2.51/Reynld*x + 0.27*Relrns)
        Deriv = 1 + 0.868589 / (x + 0.27*Relrns/2.51*Reynld)
        x = x - Fx/Deriv
End Do
Frictn = 1./(x*x)
```

Step 5: **Program control is returned back to the calling routine (sub-program PWELL()) and the sub-program FRICTN() is ended.**

```
      Return
      End
```

SUB-PROGRAM MODULE6()

LOCATION: MODULE6A.FOR

MAIN THEME: This routine is a type curve model to solve gas flow equations based on three pay grades in the reservoir. The model consists of six modules:

1. Radial flow in conventional gas reservoirs
2. Linear flow in conventional gas reservoirs (i.e. with hydraulic fractures)
3. Radial flow in naturally fractured gas reservoirs
4. Linear flow in naturally fractured gas reservoirs (i.e. with hydraulic fractures)
5. Radial flow in water drive gas reservoirs
6. Radial flow in unconventional gas reservoirs

Note that the radial flow is for vertical well and the linear flow is either for horizontal well or for hydraulic fractures.

The assumptions used in the type curve model are:

- The reservoir is initially developed based on a certain well spacing, and is produced at a constant flow rate specification for a period of time until the surface well head pressure reaches the minimum specified value.
- The gathering pressure is assumed to be the same for all wells in the reservoir.
- After the specified flow rate (for discovered reservoirs it is *gasprd93* rate and for undiscovered reservoirs it is a function of absolute open flow potential based on proration rule specified) can no longer be maintained, the reservoir is allowed to produce at a specific sandface pressure such that the well head pressure equals to the minimum specified value.
- Except for the water drive and wet unconventional (two phase) systems, a real gas potential (pseudo-pressure) approach is used as a numerical method to solve the gas equations.
- For water drive and wet unconventional systems, the common gathering pressure assumption is not used, and the production is determined using material balance approach.
- The year when primary production rate starts to decline, automatic refracturing and infill drilling are simulated in

the model and well specification is switched from flow rate to sandface pressure.

The model utilizes the following three development/production cases:

- Primary production case (no infill drilling, no refracturing)
- Refracturing initial wells (no infill drilling)
- Infill drilling, reducing the well spacing to half of its initial value

The model is designed to perform a simulation with a maximum of 41 time steps. A "speed up" option is available to get the type curve results with faster CPU time. This option uses simplified formula (coarser tolerance, etc.) and gives less accurate results.

CALLS: CALCS() (in file MODULE6D.FOR)
Performs numerical convolution for superposition in time (due to flow rate changes) and solves for pressure and flow rates at each time step to generate pressure and rates versus time.

CNTRL() (in file MODULE6D.FOR)
Initializes minimum pressures, maximum rates, skins, infill wells ON/OFF, etc.

DATOUT() (in file MODULE6D.FOR)
Prints out results to the type curve output file.

GET_TYPE() (in file MODULE6D.FOR)
Assigns number of wells, original gas in place, gas production, and well sandface pressures to type curve variables.

RDUNCN (in file MODULE6B.FOR)
Assigns sorption properties for unconventional reservoir.

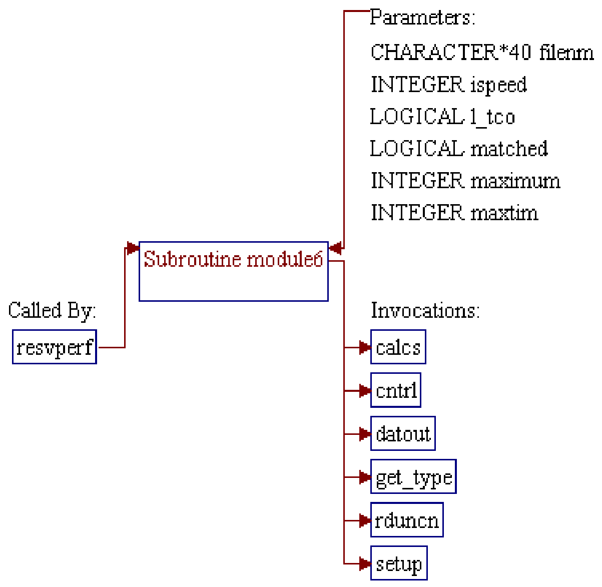
SETUP() (in file MODULE6C.FOR)
Sets up real gas potential (pseudo-pressure), viscosity, and Z-factor arrays for table lookup and calculates original gas in place.

CALLED BY: RESVPERF (in file RESVPERF.FOR)
Main program of Reservoir Performance Module.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program for the type curve model are declared. Header ".h" files are included and local variables and common blocks are defined.**

Note: Name of the sub-program is MODULE6() and the parameters passed to this sub-program are as follows:

•	*filenm*	The GSAM file name without file extension .GSM
•	*MAXTIM*	Maximum number of time steps (limited to 41 steps)
•	*ispeed*	Flag for standard formula (*ispeed=0*, higher accuracy) or simplified formula (*ispeed=1*, lower accuracy, faster)
•	*maximum*	Maximum number of development cases to be run (this value is a constant integer 3 for primary, infill, and refract)
•	*l_tco*	Logical flag whether to generate the type curve output file .TCO (.*true.*) or not (.*false.*)
•	*matched*	Logical flag whether the gas production prior to the year 1993 has been matched (.*true.*) or not (.*false.*)

```
      subroutine Module6(filenm,maxtim,ispeed,maximum,l_tco,matched)
```

Note: Global variables and common blocks are declared. Most of these declarations are stored in header files “.h”.

```
      include 'dimen.h'
      include 'welldata.h'
      include 'type_out.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
      common /stchg/iwin_yr
```

Note: Local variables are defined.

```
      logical l_tco,matched
      Character*79 Desc1$,Desc2$,FileO$
```

Step 2: **Sub-program RDUNCN is invoked.**

Note: The sub-program RDUNCN (located in file MODULE6B.FOR) obtains the unconventional reservoir informations. This routine is invoked only for the unconventional gas reservoir (module #6, *imod()* equals to 6)

```
          if(imod(1,1).eq.6) call rduncn
```

Step 3: **Name of the output file is assigned.**

Note: Output file name is the prefix of the .GSM file with extension .TCO.

```
                   FileO$ = filenm//'.tco'
```

Step 4: **Number of data points in pressure function arrays/tables (*Narray*) is set to 99.**

Note: The pressure function arrays (pseudo-pressure, viscosity, and Z-factor) are used in table lookup procedure.

```
          Narray=99
```

Step 5: **Absolute roughness of the well (*AbsRns*) is defaulted to 0.0006 inches.**

Note: The absolute roughness (*AbsRns*) of the well is used in sub-program PWELL() for friction factor calculation to determine pressure drop from the bottom hole to the well head .

```
          AbsRns=.0006
```

Step 6: **Working variables for fracture half length (*HorLen()*), fracture conductivity (*Cond()*), and pay thickness (*thickrv()*) are initialized.**

Note: In RP Module, horizontal wells are modeled as infinite conductivity horizontal fractures. In the following codes, the orientation of the well is determined from variable *Jtype()* where a value of *0* indicates a vertical well and *1* indicates a horizontal well. If the well orientation is horizontal, the length of the horizontal section of the well (*HorLen()*) is set equal to the variable *HalfLn()* and the conductivity is set to infinity (*cond() = 1E6* md-feet). Note that the *HalfLn()* is the length of the well for the horizontal well case or the fracture half length for conventional vertical well case (modeled in linear flow, Reservoir Module 2). For the case of vertical well with induced fracture, an infinite conductivity option is entered by setting a negative conductivity in input file TECH.DAT. In this case, the conductivity is again set to *1E6* md-feet. In the following section, the loop *I* is for the pay grade (1 through 3) and loop *J* is for development type (primary, infill once, infill twice (not currently used)).

```
 Do I=1, 3
         Do J=1, 3
                 If (JTyp(I,J) .eq. 1) then
                         HorLen(I,J) = HalfLn(I,J)
                         Cond(I,J)   = 1.e6
                 Else If (Cond(I,J) .le. 0.) then
                         Cond(I,J)   = 1.e6
                 End If
         End Do
    thickrv(i)=thick(i)
 End Do
```

Step 7: Sub-program SETUP() is invoked to construct pressure function tables for table lookup.

Note: The sub-program *SETUP()*, located in program file MODULE6C.FOR, generates tables for pseudo-pressure, viscosity, and Z-factor as a function of pressure to be used in table lookup routine.

```
         Call SETUP (ISpeed,IType)
```

Step 8: Loop of development cases (*ICASE*) is initialized.

Note: The RP Module generates type curves based on four development cases (*ICASE*):

- *ICASE=1* Primary development case (no infilling, no refracturing)

- *ICASE=2* Automatic refracturing when the wellhead pressure drops to or crosses the minimum pressure constraint
- *ICASE=3* Same as *ICASE=2* but with infilling once (reduce the well spacing by one-half)
- *ICASE=4* Same as *ICASE=2* but with infilling twice (reduce the well spacing by one-fourth). Note that this case is not yet implemented.

All of these development cases, except for *ICASE=4*, will be run for conventional (*ITYPE=0*) and unconventional (*ITYPE=2*) reservoirs. In the case of water drive or wet/shale reservoirs (*ITYPE=1*), the automatic refracturing case (ICASE=2) is skipped. In the following codes, the number of development cases (*MaxCas*) is first set to *maximum* (a parameter transferred from main program RESVPERF with a value of 3) and the step for the loop (*IStep*) is set to 1. For *IType=1*, the *IStep* is set to 2 to skip the automatic refracturing.

```
                  MaxCas = maximum
                  IStep  = 1
                 If (IType.eq.1) IStep  = 2
                  Do ICase = 1, MaxCas, IStep
```

Step 9: **Pay thickness (*thickrv()*) is multiplied by factor *rifact()* to take into account pay continuity as a function of well spacing.**

Note: Values for *rifact()* are set in sub-program CONVERT() using function PAYINC() (both of these sub-programs are located in file CONVERT.FOR) based on the assumption that under normal conditions, only 80% of the total pay is contacted by drilling at well spacing of 320 acres and greater. As well spacing decreases, the pay contacted increases and the *rifact()* also increases.

```
                     do ipay = 1,3
                       thick(ipay) = thickrv(ipay)*rifact(icase,ipay)
                     enddo
```

Step 10: **Sub-program CNTRL() is called to initialize minimum pressure, maximum rates, skins, infill wells on/off, etc.**

Note: The parameter *MaxTim* in sub-program CNTRL() is the maximum number of time steps (i.e. 41).

```
                     Call CNTRL (ICase, MaxTim, ISpeed)
```

Step 11: **Flag to tell whether development type change has taken place (*IChg*) and time when the change occurs (*TChg*) are initialized.**

Note: The flag *IChg* can have a value of:

- *IChg=0* Primary wells only
- *IChg=1* One set of infills
- *IChg=2* Two sets of infills (not currently used)
- *IChg=3* Primary wells after restimulation

Initial value for TChg is set to negative as an indicator that the type curve has not been run.

```
                     IChg = 0
                     TChg = -1
```

Step 12: **Sub-program CALCS() is invoked to construct pressure and rates versus time arrays.**

Note: The sub-program CALCS() performs numerical convolution for superposition in time (due to flow rate changes) and solves for pressure and flow rates at each time step (*ITime*) from the first year up to *maxtim*, to generate pressure and rates versus time.

```
                     Do ITime = 1, maxtim
                        If (ITime.le.MaxTim) then
                          Call CALCS(ITime,ICase,IChg,TChg,ISpeed,MaxTim)
                        endif
                     End Do
```

Step 13: **Sub-program DATOUT() is invoked to print out results to type curve output file.**

Note: The type curve output file (.TCO) is generated if the report is requested (*l_tco=.true.*) in input file REGIONS.DAT. Parameters *Desc1$* and *Desc2$* are string variables for header lines in output file .TCO.

```
                     if(l_tco) Call DatOut(Desc1$,Desc2$,MaxTim,
     &                 ICase,TChg)
```

Step 14: **Sub-program GET_TYPE() is invoked to get type curve variables.**

Note: The sub-program GET_TYPE() assigns number of wells, original gas in place, gas production, and well sandface pressures to type curve variables.

```
                      call get_type(maxtim,icase,tchg)
```

Step 15: **Loop of development cases (*ICASE*) is closed. The control is returned to the calling program (RESVPERF) and the sub-program is ended.**

```
                End Do
        return
        End
```

SUB-PROGRAM PD()

LOCATION: MODULE6B.FOR

MAIN THEME: This routine calculates dimensionless pressure functions based on radial flow to a well in the center of a closed square reservoir (Reservoir Module 1). These functions are used by the convolution routine to calculate pressure drops at the well, infill locations, corner location, and edge location. The corner and edge well functions are computed based on radial flow geometry (Reservoir Module 1). For Reservoir Module 2 (linear flow geometry), the functions are computed using sub-program PDWFIN() which numerically integrates the flow distribution along a fracture. Reservoir Modules 3 and 4 (naturally fractured gas reservoirs in radial and linear coordinates) implement a Warren and Root approach for fractured reservoirs in sub-program WARREN().

CALLS: EXPINT() (in file MODULE6D.FOR)
Computes exponential integral function.

PDWFIN() (in file MODULE6B.FOR)
Calculates dimensionless pressure for a well with a finite conductivity fracture producing at a constant rate in an infinite reservoir.

WARREN() (in file MODULE6C.FOR)
Calculates the difference in dimensionless pressures between a conventional reservoir and a naturally fractured reservoir using Warren and Root approach.

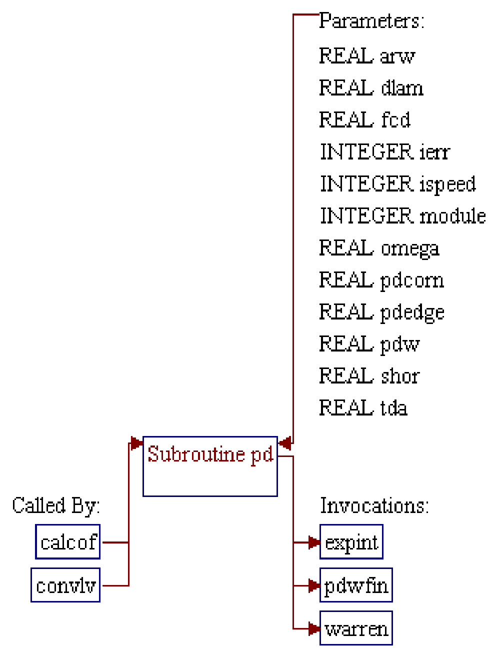
CALLED BY: CALCOF() (in file MODULE6B.FOR)
Calculates open flow potentials.

CONVLV () (in file MODULE6C.FOR)
Performs numerical convolution to determine pressure drop caused by previous production.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared.**

Note: Name of the sub-program is PD() and the parameters passed to this sub-program are as follows:

Input Paramters:

•	*Tda*	Dimensionless time
•	*Module*	Reservoir Module number
•	*ARw*	Dimensionless area factor, A**0.5/Rw, based on drainage area of primary well. XF/2 should be used instead of Rw for fractured or horizontal wells
•	*Fcd*	Dimensionless fracture conductivity for vertically fractured and horizontal wells (Reservoir Modules 2 and 4)
•	*SHor*	Equivalent skin factor for horizontal wells
•	*Omega*	Warren and Root porosity-compressibility ratio: *Omega* = $fc_{fracture} / fc_{total}$
•	*DLam*	Warren and Root interporosity flow parameter: *DLam* = $12 * k_{matrix} / k_{total} * (Rw / FracSpacing)^2$
•	*ISpeed*	Flag for standard formula (0=higher accuracy) or simplified formula (1=lower accuracy, faster)

Output Paramters:

•	*Pdw*	Dimensionless pressure at the well
•	*PdCorn*	Dimensionless pressure at the corner
•	*PdEdge*	Dimensionless pressure at the second infill location
•	*IErr*	Error flag (0=No error, 1=Error found)

```
      SUBROUTINE Pd     (Tda,    Module, ARw,    Fcd,    SHor,
     +                   Omega,  DLam,   ISpeed,
     +                   Pdw,    PdCorn, PdEdge, IErr)
```

Step 2: **Some parameters are initialized.**

```
      IErr   = 0
      Pdw    = 0.
      PdCorn = 0.
      PdEdge = 0.
```

Step 3: **Pressure functions for Reservoir Module 1 (radial flow in conventional gas reservoirs) are calculated.**

Note: Even if another Reservoir Module is entered, these calculations are used for corner and edge well pressure functions. *PdwInf* is a dimensionless pressure for a single, radial well in an infinite reservoir. One set of image wells is used for dimensionless time (*Tda*) less than 0.05. Pseudo-steady state calculation with exponential terms is used for *Tda* greater or equal to 0.05.

```
      A1 = 1./ (ARw*ARw) / ( 4.*Tda)
      PdwInf = 0.5 * ExpInt(A1)
      If (Tda .lt. 0.05) then
              A1 = 1. / ( 4.*Tda) / (ARw*ARw)
              A2 = 1. / ( 4.*Tda)
              A3 = 1. / ( 8.*Tda)
              A4 = 1. / (16.*Tda)
              A5 = 5. / (16.*Tda)
              Pdw    = 0.5 * ExpInt(A1) + 2.0 * ExpInt(A2)
              PdCorn = 2.0 * ExpInt(A3)
              PdEdge = 1.0 * ExpInt(A4) + 2.0 * ExpInt(A5)
      Else
              Pi=3.1415926
              A=4.0*Pi*Pi*Tda
              If (A .gt. 20.) A=20
              U=Exp(-A)
              Pdw    = 2.*Pi*Tda - 1.310533 + Log(ARw) - 2.0/Pi*
   +                          (U + U*U/2. + U**4/4. +U**5/5.)
              PdCorn = 2.*Pi*Tda - 0.346574 + 2.0/Pi*
   +                          (U - U*U/2. - U**4/4.)
              PdEdge = 2.*Pi*Tda - 0.173287 + 1.0/Pi*
   +                          (U*U - U**4/2.)
      End If
```

Step 4: **Pressure functions for Reservoir Module 2 or 4 (linear flow in conventional gas reservoirs or linear flow in naturally fractured gas reservoirs) is calculated.**

Note: Sub-program PDWFIN() is invoked to calculate dimensionless pressure. This routine assumes a well with finite conductivity fracture producing at a constant rate in an infinite reservoir.

```
      If ((Module .eq. 2) .or. (Module .eq. 4)) then
              Tdxf = Tda * ARw * ARw / 4.
              if(Tdxf.le.0.)Tdxf=0.
              P    = PdwFin (Tdxf, Fcd, SHor, ISpeed)
              Pdw  = Pdw + P – PdwInf
      End If
```

Step 5: **Pressure functions for Reservoir Module 3 or 4 (radial flow in naturally fractured gas reservoirs or linear flow in naturally fractured gas reservoirs) is corrected with Warren and Root routine.**

Note: Pressure functions for Reservoir Modules 3 and 4 were calculated in Step 3 and Step 4, respectively. Sub-program WARREN() is invoked to calculate correction to pressure functions due to fracture using Warren and Root approach. This correction is added to the previously calculated pressure functions (*Pdw*).

```
If ((Module .eq. 3) .or. (Module .eq. 4)) then
        Tdw   = Tda * ARw * ARw / 4.
        Pnatf = Warren(Tdw,Omega,DLam)
        Pdw   = Pdw + Pnatf
End If
```

Step 6: **Error flag is set to 1 if Module number is out of range.**

```
If ((Module .lt. 1) .or. (Module .gt. 6)) IErr = 1
```

Step 7: **Error flag is set to 1 and dimensionless pressures are set to big numbers (1000000) if dimensionless time is negative.**

```
If (Tda.le.0.) then
        IErr   = 1
        Pdw    = 1000000.
        PdCorn = 1000000.
        PdEdge = 1000000.
End If
```

Step 8: **Program control is returned back to the calling routines (sub-program CALCOF() or CONVLV()) and the sub-program PD() is ended.**

```
Return
End
```

SUB-PROGRAM PDWFIN()

LOCATION: MODULE6B.FOR

MAIN THEME: This routine calculates dimensionless pressure for a well with a finite conductivity fracture producing at a constant rate in an infinite reservoir. The solution involves matching the pressure drop at a point along the fracture, using the Uniform Flux Solution from Gringarten, et al. (1974). The calculation point is based on a correlation from Blasingame and Poe. The results were essentially the same as those of Wong, et al. for TDXF*FCD*FCD > 0.1. Horizontal wells are modeled based on an equivalent skin factor applied to the dimensionless pressure of the vertical fracture, using the analytical skin factor formula presented by Ozkan, Raghavan, and Joshi, SPE Formation Evaluation, Dec., 1989, pp 567-575.

CALLS: ERRFN() (in file MODULE6D.FOR)
Computes error function.

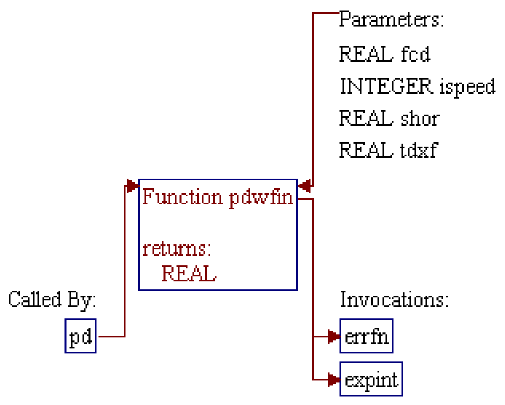
EXPINT() (in file MODULE6D.FOR)
Computes exponential integral function.

CALLED BY: PD() (in file MODULE6B.FOR)
Calculates dimensionless pressure functions for different reservoir systems.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared.**

Note: Name of the sub-program is PDWFIN() and the parameters passed to this sub-program are as follows:

Input Parameters:

•	*Tdxf*	Dimensionless time based on fracture half length XF, TDXF=KT/PHI/MU/C/XF/XF
•	*Fcd*	Dimensionless fracture conductivity, FCD=KFW/K/XF
•	*SHor*	Equivalent skin factor for horizontal wells
•	*ISpeed*	Flag for standard formula (0=higher accuracy) or simplified formula (1=lower accuracy, faster)

Output Parameter:

•	*PdwFin*	Dimensionless pressure at the well

```
      FUNCTION   PdwFin (Tdxf,   Fcd,    SHor,   ISpeed)
```

Step 2: **Constants for arguments of the error and exponential integral functions are assigned.**

```
      Data a0,a1,a2,a3,a4,b0,b1,b2,b3,b4/
     +        0.759919, 0.465301, 0.562754, 0.363093, 0.029881,
     +        1.000000, 0.994770, 0.896679, 0.430707, 0.0467339/
```

Step 3: **Arguments for the error and exponential integral functions are calculated.**

```
      if(tdxf.le.0.0)tdxf = 0.0001
      F = Fcd
      If (F .lt. 0.5)  F = 0.5
      If (F .gt. 500.) F = 500.
      C  = Log(F)
      X = (a0 + C * (a1 + C * (a2 + C * (a3 + C * a4)))) /
     +    (b0 + C * (b1 + C * (b2 + C * (b3 + C * b4))))
      Arg1 = (1.-X) / 2. / Sqrt(TdXf)
      Arg2 = (1.+X) / 2. / Sqrt(TdXf)
      Arg3 = Arg1 * Arg1
      Arg4 = Arg2 * Arg2
```

Step 4: **Sub-programs ERRFN() and EXPINT() are invoked to calculate error function and exponential integral of the arguments in Step 3.**

```
      C1 = ErrFn (Arg1)
```

```
        C2 = ErrFn (Arg2)
        C3 = ExpInt(Arg3)
        C4 = ExpInt(Arg4)
```

Step 5: **Dimensionless pressure is calculated.**

```
        PdwFin = (C1+C2) * Sqrt(3.1415926*TdXf)/2. +
     +           (1.-X) * C3 / 4. + (1.+X) * C4 / 4. + SHor
```

Step 6: **Program control is returned back to the calling routine (sub-program PD ()) and the sub-program PDWFIN() is ended.**

```
        Return
        End
```

SUB-PROGRAM RATE1()

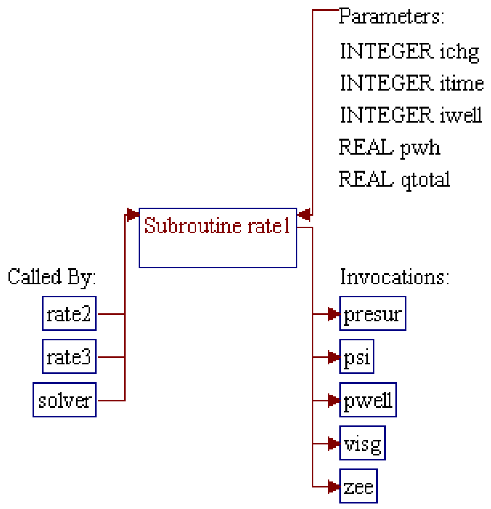
LOCATION:	MODULE6B.FOR
MAIN THEME:	This routine calculates gas flow rates for primary well under pressure constraint.

CALLS:	PRESUR() (in file MODULE6B.FOR)
	Performs inverse table look-up of pressure given real gas potential (psudo-pressure).
	
	PSI() (in file MODULE6B.FOR)
	Performs table look-up of real gas potential (psudo-pressure) given pressure.
	
	PWELL() (in file MODULE6B.FOR)
	Calculates bottomhole pressure, wellhead pressure, or flow rate based on the difference between well head pressure and bottom hole pressure of the well using Smith's formula.
	
	VISG() (in file MODULE6C.FOR)
	Performs table look-up of gas viscosity as a function of pressure using linear interpolation.
	
	ZEE() (in file MODULE6C.FOR)
	Performs table look-up linear interpolation of Z-factor as a function of pressure.

CALLED BY:	RATE2() (in file MODULE6B.FOR)
	Calculates flow rates of primary wells based on rate constraint. Calculates each pay grade separately using same wellhead pressure.
	
	RATE3() (in file MODULE6B.FOR)
	Calculates gas flow rates for infill wells (infill once) under pressure constraint.
	
	SOLVER() (in file MODULE6C.FOR)
	Solves for flow rates and pressures within specified time step.

READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included.**

Note: Name of the sub-program is RATE1() and the parameters passed to this sub-program are as follows:

•	*Pwh*	Wellhead pressure (psia)
•	*QTotal*	Total gas production from field (MCF/D)
•	*ITime*	Time step number
•	*IChg*	Flag to indicate whether development type change has taken place: 0=primary wells only, 1=one set of infills, 2=two sets of infills (not currently used), 3= primary wells after restimulation
•	*IWell*	Flag to indicate well type: 1=primary well, 2=infill well

```
      SUBROUTINE Rate1  (Pwh,    QTotal, ITime,  IChg,  IWell)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type10.h'
```

Step 2: **Loop for pay grades (loop *J*) is initialized.**

```
      Do J = 1, 3
```

Step 3: **Gas and water production rates, *Qg()* and *Qw()*, are initialized.**

Note: The gas and water production rates use three-parameter array variables *Qg(p1,p2,p3)* and *Qw(p1,p2,p3)*, where *p1* is for pay grades (1,2,3), *p2* is for primary or infill (1=primary, 2=infill), and *p3* is for time step (1,2,3,...). The following codes set the values of gas and water flow rates of infill wells equal to zeros.

```
                Qg(J,2,ITime) = 0.
                Qw(J,2,ITime) = 0.
                Qg(J,3,ITime) = 0.
                Qw(J,3,ITime) = 0.
```

Step 4: **The gas and water flow rates of the corresponding pay grade are set to zeros if the wells are shut in (*KShut()>0*).**

```
                If (KShut(J).gt.0) then
                    Qg(J,IWell,ITime) = 0.
                    Qw(J,IWell,ITime) = 0.
```

Step 5: **The gas and water flow rates calculations are performed if the wells are on production (*KShut()=0*). The first step is obtaining information on skin factor (*S*) and well depth (*Dep*).**

Note: The skin factor is obtained from three-parameter array variable *Skin(p1,p2,p3)* where:

- *p1* Parameter for pay grade (1,2,3)
- *p2* Parameter for primary or infill wells (1=primary, 2=infill)
- *p3* Parameter for stimulation type (1=no refrac, 2=with refrac)

Note that parameters *p2* and *p3* are dependent on each other because the current version of the RP Module does not consider infilling with refracturing as a development case. Therefore, for *p2=2*, the *p3* can only have a value of *1*. The well depth is obtained from variable *Depth1()*.

```
                Else
                    S = Skin(J,IWell,1)
                    If (IChg.eq.3) S = Skin(J,IWell,2)
                    Dep = Depth1(J)
```

Step 6: **Sub-programs PWELL() and PSI() are invoked to calculate minimum bottomhole pressure (*PbhMin*) and minimum pseudo-pressure (*PsiMin*).**

Note: The calculations based on hydrostatic pressure (flow rate is set to zero). Well type (*KTyp*) is obtained from variable (*KUnCon()*) which is type of unconventional reservoir (0=dry coal, 1=wet coal, 2=dry shale, 3=wet shale).

```
                  QMin = 0.
                  KTyp = KUnCon(J)
                  Call PWELL(Pwh,PbhMin,QMin,Deriv,Dep,1,IErr,KTyp,J)
                  PsiMin = Psi(PbhMin,NArray,PreAry,PsiAry)
```

Step 7: **Maximum bottomhole pressure (*PbhMax*) is calculated.**

Note: *PbhMax* is calculated based on initial reservoir pressure (*PInit()*) in sub-program PRESSURE().

```
                  PsiC=Psi(PInit(J),NArray,PreAry,PsiAry)
                  If (ITime .gt. 1) then
                     PsiC = PsiC - DPsi(J,IWell)
                     Do L = 1, 3
                       PsiC=PsiC+PsiCon(J)*C(J,IWell,L)*Qg(J,L,ITime-1)
                     End Do
                  End If
                  PbhMax = Presur(PsiC,NArray,PreAry,PsiAry)
```

Step 8: **Maximum flow rate (*Q*) is set to zero if the minimum bottomhole pressure is higher than the maximum bottomhole pressure.**

```
                  If (PbhMin .gt. PbhMax) then
                    Q = 0
```

Step 9: **Maximum flow rate (*Q*) is determined if the minimum bottomhole pressure is at the most equal to the maximum bottomhole pressure.**

Note: The first alternative for maximum flow rate (*QMax1*) is based on the maximum bottomhole pressure calculated in sub-program PWELL(). The second alternative (*QMax2*) is based on reservoir flow against the calculated minimum pseudo-pressure (*PsiMin*) at the wellbore. The third alternative (*QMax3*) is based on maximum recovery.

```
                  Else
                    Call PWELL(Pwh,PbhMax,QMax1,Deriv,Dep,3,IErr,KTyp,J)
                    QMax2=(PsiC-PsiMin)/(PsiCon(J)*(C(J,IWell,IWell)+S))
                    DT = Time(ITime)
                    If (ITime.gt.1) DT = DT - Time(ITime-1)
                    Pi   = Pinit(J)
                    Zi   = Zee(Pi,NArray,PreAry,ZAry)
                    P1   = PreMin
                    Z1   = Zee(P1,NArray,PreAry,ZAry)
                    REMax = 1.-(P1/Z1)/(Pi/Zi)
                    QMax3 = (OGIP1(J)*REMax-CumGas(J,1,ITime)-
     +                 CumGas(J,2,ITime)-2.*CumGas(J,3,ITime))/(365.*DT)
```

Note: Maximum flow rate (*Q*) is set to zero and the well is shut in if the *QMax3* is too small (less than 1 MCF/well).

```
                     If (QMax3.le.1.) then
                       KShut(J) = 1
                       Q = 0.
```

Note: The maximum flow rate is taken as the lowest of the three maximum rate alternatives and the minimum value is set to be 100 MCF/well. Based on this maximum flow rate (store in variable *QMx*), maximum bottomhole pressure (*PbhMax*) is recalculated in sub-program PWELL().

```
                     Else
                       QMx = Min(QMax1, QMax2, QMax3)
                       QMx = Max(QMx, 100.)
                       Call PWELL(Pwh,PbhMax,QMx,Deriv,Dep,1,IErr,KTyp,J)
```

Note: The flow rate as a function of bottomhole pressure is fitted to a quadratic equation. From this fitted equation, maximum flow rate (*Q*) is determined and the minimum value is again set to 100 MCF/well.

```
                       PsiMax = Psi(PbhMax,NArray,PreAry,PsiAry)
                       A      = PsiMin
                       B      =(PsiMax - A) / QMx ** 2
                       AQ     = B
                       BQ     = PsiCon(J) * (C(J,IWell,IWell) + S)
                       CQ     = A - PsiC
                       Disc   = BQ ** 2 - 4. * AQ * CQ
                       Disc   = Max(Disc,0.)
                       SD     = Sqrt(Disc)
                       If ((200.*AQ) .lt. (SD-BQ)) then
                          Q = ( SD – BQ ) / (2. * AQ)
                       Else
                         Q = 100.
                       End If
                       Q = Max(Q, QMx)
```

Note: A Newton-Raphson iterative procedure with a maximum of 10 iterations is performed to determined the maximum flow rate (*Q*). This iteration will yield a consistent bottomhole pressure and flow rate within a flow rate tolerance of 0.1 MCF/well. The resulting gas flow rate is then stored in type curve variable *Qg()*.

```
                      ISolve = 0
                       Do Iter = 1, 10
                         If (ISolve .eq. 0) then
                             Psi1 = PsiC-PsiCon(J)*(C(J,IWell,IWell)+S)*Q
                             Pbh1 = Presur(Psi1,NArray,PreAry,PsiAry)
                             Call PWELL(Pwh,Pbh2,Q,Deriv,Dep,1,
     +                                      IErr,KTyp,J)
```

```
                             Psi2 = Psi(Pbh2,NArray,PreAry,PsiAry)
                             F  = Psi2 - Psi1
                             DP = Pbh1 - Pbh2
                             Z  = Zee(Pbh2,NArray,PreAry,ZAry)
                             V  = Visg(Pbh2,NArray,PreAry,VisAry,Va)
                             FP = 2. * Pbh2 / (V * Z) / Deriv +
     +                               PsiCon(J) * (C(J,IWell,IWell)+S)
                             DelRat = F/FP
                             If (Abs(DelRat).lt.0.1) then
                                 ISolve = Iter
                             Else
                                 QT = Q-DelRat
                                 If (QT .lt. Q/2.)  QT=Q/2.
                                 If (QT .gt. Q*2.)  QT=Q*2.
                                 If (QT .gt. QMax3) QT=QMax3
                                 Q = QT
                             End If
                         End If
                       End Do
                   End If
                 End If
                 Qg(J,IWell,ITime) = Q
             End If
         End Do
```

Step 10: **Total flow rate from the three pay grades *(QTotal)* is calculated.**

```
         QTotal = 0.
         Do I=1,3
                 QTotal = QTotal + Area(I) / WSpace(I) *
     +               (Qg(I,1,ITime) + Qg(I,2,ITime) + Qg(I,3,ITime)*2.)
         End Do
```

Step 11: **The program control is returned back to the calling routine (sub-program RATE2(), RATE3(), or SOLVER()) and the sub-program RATE1() is ended.**

```
         Return
         End
```

SUB-PROGRAM RATE2()

LOCATION: MODULE6B.FOR

MAIN THEME: This routine calculates flow rates of primary wells based on rate constraint. It calculates each pay grade separately using same wellhead pressure.

CALLS: RATE1() (in file MODULE6B.FOR)
Calculates gas flow rates for primary wells under pressure constraint.

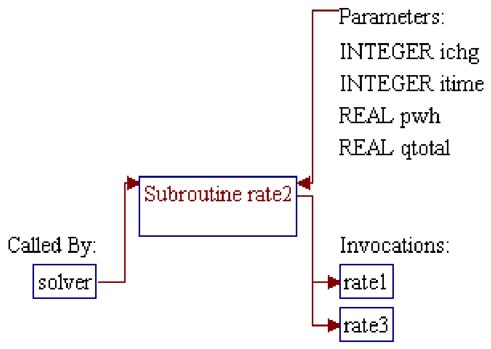
RATE2() (in file MODULE6B.FOR)
Calculates flow rates of primary wells based on rate constraint. Calculates each pay grade separately using same wellhead pressure.

CALLED BY: SOLVER() (in file MODULE6C.FOR)
Solves for flow rates and pressures within specified time step.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included. Local variable is declared.**

Note: Name of the sub-program is RATE2() and the parameters passed to this sub-program are as follows:

Pwh	Wellhead pressure (psia)
QTotal	Total gas production from field (MCF/D)
ITime	Time step number
IChg	Flag to indicate whether development type change has taken place: 0=primary wells only, 1=one set of infills, 2=two sets of infills (not currently used), 3= primary wells after restimulation

```
      SUBROUTINE Rate2  (Pwh,    QTotal, ITime,  IChg)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type10.h'
```

Note: Local variable as a working variable for shut in a well is declared.

```
      Dimension JShut(3)
```

Step 2: **Perform 3 Bi-section iterations to get good estimate of pressure for Newton-Raphson iteration.**

Note: Assign minimum and maximum pressures for Bi-section iteration. For the first time step (ITime=1) the maximum pressure (*P1* or *Pmax*) is set equal to the highest initial pressure from the three pay grades. For time step greater than 1, the highest average reservoir pressure from the previous time step is used as maximum pressure. The minimum pressure (*P2* or *Pstart*) is obtained from global variable *PreMin*.

```
         Pmax = Max(Pinit(1), Pinit(2), Pinit(3))
         If (ITime.gt.1) Pmax = Max(PreAvg(1,ITime-1), PreAvg(2,ITime-1),
      +                    PreAvg(3,ITime-1))
         Pstart = PreMin
         P1 = Pmax
         P2 = Pstart
```

Note: Perform 3 Bi-section iterations to get initial guess pressure *(P)* for Newton-Raphson iteration. In this iteration, sub-program RATE1() is invoked to calculate well flow rates (based on pressure constraint) if the system is primary well or primary well after restimulation (*IChg=0* or *3*). For infill wells (*IChg=1* or *2*), flow rate is calculated in sub-program RATE3() (based on rate constraint).

```
         JShut(1) = KShut(1)
         JShut(2) = KShut(2)
         JShut(3) = KShut(3)
         Do Iter = 1, 3
             KShut(1) = JShut(1)
             KShut(2) = JShut(2)
             KShut(3) = JShut(3)
             P    = (Pstart + PMax) / 2.
             If ((IChg .eq. 0).or.(IChg .eq. 3)) then
                 IWell = 1
                 Call RATE1 (P,QTotal,ITime,IChg,IWell)
             Else If ((IChg .eq. 1).or.(IChg .eq. 2)) then
                 Call RATE3 (P, QTotal, ITime, IChg)
             End If
             F = QTotal - RatMax
             If (F.gt.0.) then
                 Pstart = P
             Else
                 Pmax = P
             End If
         End Do
```

Step 3: **Perform a maximum of 10 Newton-Raphson iterations to determin gas flow rates.**

Note: Initial pressure for the Newton-Raphson procedure is taken as the middle point between pressures *Pstart* and *PMax* from the Bi-secdtion iteration. Epsilon pressure for numerical integration (*DelP*) is set equal to 2 psi and convergence flag (*ISolve*) is set to zero to indicate non-convergence condition.

```
         P    = (Pstart + PMax) / 2.
         DelP = 2.
         ISolve = 0
```

Note: Newton-Raphson iteration is performed. Sub-program RATE1() or RATE3() are invoked twice, first at pressure *P* and second at pressure *P+DelP*. Backward difference is then used to approximate derivative of the objective function with respect to pressure. A flow rate tolerance of 0.0001 MCFD is used as convergence criterion. After the convergence is achieved, the resulting pressure (*P*) is assigned to the wellhead pressure (*Pwh*).

```
Do Iter = 1, 10
    If (ISolve .eq. 0) then
        KShut(1) = JShut(1)
        KShut(2) = JShut(2)
        KShut(3) = JShut(3)
        If ((IChg .eq. 0).or.(IChg .eq. 3)) then
            IWell = 1
            Call RATE1 (P,QTotal,ITime,IChg,IWell)
        Else If ((IChg .eq. 1).or.(IChg .eq. 2)) then
            Call RATE3 (P, QTotal, ITime, IChg)
        End If
        F = QTotal - RatMax
        If (Abs(F) .lt. 0.1) then
            ISolve=Iter
        Else
            P1 = P + DelP
            If ((IChg .eq. 0).or.(IChg .eq. 3)) then
                IWell = 1
                Call RATE1 (P1,QTot1,ITime,IChg,IWell)
            Else If ((IChg.eq.1).or.(IChg.eq.2)) then
                Call RATE3 (P1,QTot1,ITime,IChg)
            End If
            FP = QTot1 - RatMax - F
            If (Abs(FP) .gt. 0.0001) then
                PG = P - F/FP * DelP
                If (PG.lt.P/2.) PG = P/2.
                If (PG.gt.PMax) PG = (P+PMax)/2.
                If (Abs(FP).lt.0.02) DelP =Min(2.*DelP,16.)
                P = PG
            End If
        End If
    End If
End Do
Pwh = P
```

Step 4: **The program control is returned back to the calling routine (sub-program SOLVER()) and the sub-program RATE2() is ended.**

```
Return
End
```

SUB-PROGRAM RATE3()

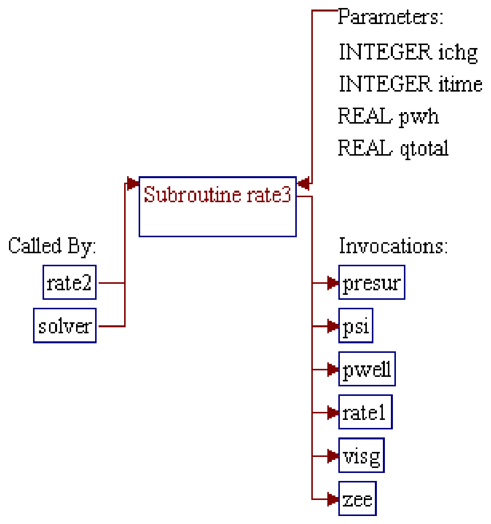
LOCATION:	MODULE6B.FOR
MAIN THEME:	This routine calculates gas flow rates for infill wells (infill once) under pressure constraint.

CALLS:	PRESUR() (in file MODULE6B.FOR)
	Performs inverse table look-up of pressure given real gas potential (psudo-pressure).
	
	PSI() (in file MODULE6B.FOR)
	Performs table look-up of real gas potential (psudo-pressure) given pressure.
	
	PWELL() (in file MODULE6B.FOR)
	Calculates bottomhole pressure, wellhead pressure, or flow rate based on the difference between well head pressure and bottom hole pressure of the well using Smith's formula.
	
	RATE1() (in file MODULE6B.FOR)
	Calculates gas flow rates for primary wells under pressure constraint.
	
	VISG() (in file MODULE6C.FOR)
	Performs table look-up of gas viscosity as a function of pressure using linear interpolation.
	
	ZEE() (in file MODULE6C.FOR)
	Performs table look-up linear interpolation of Z-factor as a function of presure.

CALLED BY:	RATE2() (in file MODULE6B.FOR)
	Calculates flow rates of primary wells based on rate constraint. Calculates each pay grade separately using same wellhead pressure.
	
	SOLVER() (in file MODULE6C.FOR)
	Solves for flow rates and pressures within specified time step.

READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included. Local variables are declared.**

Note: Name of the sub-program is RATE3() and the parameters passed to this sub-program are as follows:

• *Pwh*	Wellhead pressure (psia)
• *QTotal*	Total gas productio n from field (MCF/D)
• *ITime*	Time step number
• *IChg*	Flag to indicate whether development type change has taken place: 0=primary wells only, 1=one set of infills, 2=two sets of infills (not currently used), 3= primary wells after restimulation

```
      SUBROUTINE Rate3  (Pwh,    QTotal, ITime,  IChg)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type10.h'
```

Note: Local variables are declared.

```
      Dimension QChk(3,3), JShut(3)
```

Step 2: **Values of variable *KShut()* to indicate whether the area is shut in, are stored to working array *JShut()*.**

```
      JShut(1) = KShut(1)
      JShut(2) = KShut(2)
      JShut(3) = KShut(3)
```

Step 3: **Sub-program RATE1() is invoked twice to calculate gas flow rates for primary wells at minimum pressure and for infill wells at wellhead pressure.**

Note: The resulting gas flow rates for both primary and infill wells are stored in working variable *QChk()*. At the end of these calculations, all gas flow rates (*Qg()*) are reset to zeros. Note that the gas rates use three-parameter array variables *Qg(p1,p2,p3)* where *p1* is for pay grades (1,2,3), *p2* is for primary or infill (1=primary, 2=infill), and *p3* is for time step (1,2,3,...). In the following codes, variable *IWell* denotes primary wells (*IWell=1*) or infill wells (*IWell=2*).

```
IWell = 1
Call RATE1 (PreMin, QTotal, ITime, IChg, IWell)
Do I = 1, 3
        QChk(I,IWell) = Qg(I,IWell,ITime)
End Do
IWell = 2
Call RATE1 (Pwh, QTotal, ITime, IChg, IWell)
Do I = 1, 3
        QChk(I,IWell) = Qg(I,IWell,ITime)
        Do J = 1, 3
                Qg(I,J,ITime) = 0.
        End Do
End Do
```

Step 4: **Time step size, the difference between two consecutive time levels, is set.**

```
DT = Time(ITime)
If (ITime.gt.1) DT = DT - Time(ITime-1)
```

Step 5: **Flow rate *QChk()* is updated if the value is higher than flow rate based on maximum recovery (*QMax3*).**

```
         Do J = 1, 3
             Pi   = Pinit(J)
             Zi   = Zee(Pi,NArray,PreAry,ZAry)
             P1   = PreMin
             Z1   = Zee(P1,NArray,PreAry,ZAry)
             REMax = 1.-(P1/Z1)/(Pi/Zi)
             QMax3 = (OGIP1(J)*REMax-CumGas(J,1,ITime)-CumGas(J,2,ITime)-
   +               2.*CumGas(J,3,ITime)) / (365.*DT)
             If (QChk(J,1).gt.QMax3) QChk(J,1) = QMax3
             If (QChk(J,2).gt.QMax3) QChk(J,2) = QMax3
```

Step 6: **The well is shut in (*KShut=1*) if flow rate from maximum recovery is too low (less than 1 MCF/well).**

```
    If (QMax3.le.1.) then
        KShut(J) = 1
```

Step 7: **Calculations for producing reservoirs (*KShut()=0*) are performed.**

Note: Well constants such as well depth (*Dep*), type of unconventional reservoir (*KTyp*) and skin factors (*S1* for primary wells and *S* for infill wells) are set.

```
            Else If (JShut(J).eq.0) then
                Dep  = Depth1(J)
                KTyp = KUnCon(J)
                S1   = Skin(J,1,1)
                S    = Skin(J,2,1)
```

Note: Sub-program PSI() is invoked to convert initial pressure into pseudo-pressure (real gas potential).

```
                Psi1 = Psi(PInit(J),NArray,PreAry,PsiAry)
                Psi2 = Psi1
```

Note: Sub-program PSI() is invoked to convert initial pressure into pseudo-pressure (real gas potential). For the first time step (*ITime=1*), the pseudo-pressure for infill wells (*Psi2*) is set equal to pseudo-pressure of primary wells (*Psi1*). For the next time step (*ITime>1*) the pseudo-pressures are updated with pseudo-pressure drops due to productions, the terms with *C(p1,p2,p3)*, where *p1* is for pay grade, *p2* is for primary or infill wells, and *p3* is interference term for other wells (1=corner of primary, 2=corner of infill, 3=edge).

```
                If (ITime.gt.1) then
                    Psi1 = Psi1 - DPsi(J,1) + PsiCon(J) *
     +              (C(J,1,1)*Qg(J,1,ITime-1)+C(J,1,2)*Qg(J,2,ITime-1))
                    Psi2 = Psi2 - DPsi(J,2) + PsiCon(J) *
     +              (C(J,2,1)*Qg(J,1,ITime-1)+C(J,2,2)*Qg(J,2,ITime-1))
                End If
```

Note: Make sure wellhead pressure for primary well is higher than minimum pressure (*PreMin*). If not, shut the well in.

```
                Q = 0.
                Psi1a  = Psi1 - PsiCon(J) * C(J,1,2) * QChk(J,2)
                Pbh1a  = Presur(Psi1a,NArray,PreAry,PsiAry)
                Call PWELL(Pwh1a,Pbh1a,Q,Deriv,Dep,2,IErr,KTyp,J)
                If ((Pwh1a.le.Premin).or.(QChk(J,1).le.0.)) then
                  Qg(J,1,ITime) = 0.
                  Qg(J,2,ITime) = Max(QChk(J,2),0.)
                  Qg(J,3,ITime) = 0.
```

Note: or if maximum rate form infill well is less than zero, shut the well in

```
        Else If (QChk(J,2).le.0.) then
          Qg(J,1,ITime) = Max(QChk(J,1),0.)
          Qg(J,2,ITime) = 0.
          Qg(J,3,ITime) = 0.
```

Note: or reduces wellhead pressure of infill to *Pwh* and primary well pressure to *PreMin* then check whether the wellhead pressure is greater than pressure of infill well at zero rate.

```
        Else
          Q = 0.
          Psi2a = Psi2 - PsiCon(J) * C(J,2,1) * QChk(J,1)
          Pbh2a = Presur(Psi2a,NArray,PreAry,PsiAry)
          Call PWELL(Pwh2a,Pbh2a,Q,Deriv,Dep,2,IErr,KTyp,J)

          If (Pwh2a.le.Pwh) then
            Qg(J,1,ITime) = Max(QChk(J,1),0.)
            Qg(J,2,ITime) = 0.
            Qg(J,3,ITime) = 0.
          Else
```

Note: Calculate constants for quadratic fit of bottomhole pressure versus flow rate. Use this equation to calculate flow rate to be used as initial guess for Newton-Raphson iteration.

```
            Q = 0.
            Call PWELL(PreMin,PbhMn1,Q,Deriv,Dep,1,IErr,KTyp,J)
            A1 = Psi(PbhMn1,NArray,PreAry,PsiAry)
            Call PWELL(Pwh,PbhMn2,Q,Deriv,Dep,1,IErr,KTyp,J)
            A2 = Psi(PbhMn2,NArray,PreAry,PsiAry)
            Q = Max(QChk(J,1),100.)
            Call PWELL(PreMin,PbhMx1,Q,Deriv,Dep,1,IErr,KTyp,J)
            B1 = (Psi(PbhMx1,NArray,PreAry,PsiAry)-A1) / Q**2
            Q = Max(QChk(J,2),100.)
            Call PWELL(PreMin,PbhMx2,Q,Deriv,Dep,1,IErr,KTyp,J)
            B2 = (Psi(PbhMx2,NArray,PreAry,PsiAry)-A2) / Q**2
            A1 = A1 - Psi1
            A2 = A2 - Psi2
            C1 = PsiCon(J) * (C(J,1,1)+S1)
            C2 = PsiCon(J) * (C(J,2,2)+S)
            D1 = PsiCon(J) *  C(J,1,2)
            D2 = PsiCon(J) *  C(J,2,1)
            CO0 = (B1*A2**2    + A1*D2**2 - A2*C1*D2)/(B1*B2**2)
            CO1 = (2.*A2*B1*C2 + D1*D2**2 - C1*C2*D2)/(B1*B2**2)
            CO2 = (2.*A2*B1*B2 + B1*C2**2 - B2*C1*D2)/(B1*B2**2)
            CO3 = (2.*B1*B2*C2)/(B1*B2**2)
            DP = (C1+D1)**2 - 4.*B1*A1
            If (DP.ge.0.) then
                Q1 = (-(C1+D1)+sqrt(DP))/(2.*B1)
            Else
                Q1 = A1/(C1+D1)
            End If
            Q1 = Max(Q1, 100.)
            DI = (C2+D2)**2 - 4.*B2*A2
            If (DI.ge.0.) then
                Q2 = (-(C2+D2)+sqrt(DI))/(2.*B2)
            Else
```

```
                Q2 = A2/(C2+D2)
            End If
            Q2 = Max(Q2, 100.)
```

Note: Use Newton-Raphson to solve for flow rates. A two-dimensional iterative solution is set up to determine *Q1* and *Q2* (the rates at primary and infill wells, respectively) that cause the bottomhole pseudo-pressures at the wells to be the same whether computed from reservoir pressure drop at the iterated rates, or from wellbore pressure change.

```
            ISolve = 0
            Do Iter = 1, 15
                If (ISolve .eq. 0) then
                    Psi1a = Psi1-PsiCon(J)*
     +                  ((C(J,1,1)+S1)*Q1+C(J,1,2)*Q2)
                    Pbh1 =Presur(Psi1a,NArray,PreAry,PsiAry,I0)
                    Call PWELL(PreMin,Pbh2,Q1,Deriv1,Dep,1,
     +                           IErr,KTyp,J)
                    Psi1b = Psi(Pbh2,NArray,PreAry,PsiAry)
```

Note: Calculate the following:

• *F*	difference in bottomhole pseudo-pressure for primary well
• *FQ1*	partial derivative of *F* with respect to *Q1*
• *FQ2*	partial derivative of *F* with respect to *Q2*
• *G*	difference in bottomhole pseudo-pressure for infill well
• *GQ1*	partial derivative of *G* with respect to *Q1*
• *GQ2*	partial derivative of *G* with respect to *Q2*

```
                    F = Psi1b - Psi1a
                    Z = Zee(Pbh2,NArray,PreAry,ZAry)
                    V = Visg(Pbh2,NArray,PreAry,VisAry,Va)
                    Fq1 = 2.*Pbh2/(V*Z)/Deriv1 +
     +                      PsiCon(J)*(C(J,1,1)+S1)
                    Fq2 = PsiCon(J) * C(J,1,2)
                    DP1 = Pbh1 - Pbh2
                    Psi2a = Psi2-PsiCon(J)*
     +              ((C(J,2,2)+S)*Q2+C(J,2,1)*Q1)
                    Pbh1 =Presur(Psi2a,NArray,PreAry,PsiAry,I0)
                    Call PWELL(Pwh,Pbh2,Q2,Deriv2,Dep,1,
     +                           IErr,KTyp,J)
                    Psi2b = Psi(Pbh2,NArray,PreAry,PsiAry)
                    G = Psi2b - Psi2a
                    Z = Zee(Pbh2,NArray,PreAry,ZAry)
                    V = Visg(Pbh2,NArray,PreAry,VisAry,Va)
                    Gq2 = 2.*Pbh2/(V*Z)/Deriv2 +
     +                      PsiCon(J) * (C(J,2,2)+S)
                    Gq1 = PsiCon(J) * C(J,2,1)
                    DP2 = Pbh1 - Pbh2
```

Note: Solve within 0.1 MCFD

```
                            DelQ1 = (F*Gq2-G*Fq2)/(Fq1*Gq2-Fq2*Gq1)
                            DelQ2 = (Fq1*G-Gq1*F)/(Fq1*Gq2-Fq2*Gq1)
                            ITest = 0
                            If ((abs(DelQ1).lt.0.1) .and.
     +                          (abs(DelQ2).lt.0.1)) ITest = 1
                            If (ITest.eq.1) then
                                ISolve = Iter
                            Else
                                R1=DelQ1/Q1
                                R2=DelQ2/Q2
                                R = 1.
```

Note: Now check for cases where infill well pressure will be below the specified pressure. In such case, let the infill well rate be less than 1 MCFD and do a one-dimensional Newton-Raphson on the primary well rate.

```
                                If ((R2.gt.5.).and.(Q2.lt.0.3)) then
                                    DelRat = F/Fq1
                                    QT=Q1-DelRat
                                    If (QT.lt.Q1/2.) QT = Q1/2.
                                    If (QT.gt.Q1*2.) QT = Q1*2.
                                    Q1 = QT
                                    If (abs(DelRat).lt.0.01) then
                                        ISolve = 1
                                        Q1 = QChk(J,1)
                                        Q2 = 0.
                                    End If
                                Else
```

Note: Otherwise, continue the two-dimensional Newton-Raphson solution.

```
                                    If (R1.gt.0.5) R = 0.5/R1
                                    If (R2.gt.0.5) R = Min(R,0.5/R2)
                                    If (R1.lt.-1.) R = Min(R,-1./R1)
                                    If (R2.lt.-1.) R = Min(R,-1./R2)
                                    Q1 = Q1-DelQ1*R
                                    Q2 = Q2-DelQ2*R
                                End If
                            End If
                        End If
                    End Do
```

Note: Check flow rates versus maximum recovery efficiency. Adjust flow rates if necessary.

```
                    If (Q1+Q2 .gt. QMax3) then
                        Q =  Q1 + Q2
                        Q1 = Q1 / Q * QMax3
                        Q2 = Q2 / Q * QMax3
                    End If
                    Qg(J,1,ITime) = Q1
                    Qg(J,2,ITime) = Q2
                    Qg(J,3,ITime) = 0.
                  End If
                End If
            End If
```

```
      End Do
```

Step 8: **Total gas production (*QTotal*) is calculated.**

```
      QTotal = 0.
      Do I=1,3
              QTotal = QTotal + Area(I) / WSpace(I) *
   +                 (Qg(I,1,ITime) + Qg(I,2,ITime) + Qg(I,3,ITime)*2.)
      End Do
```

Step 9: **The program control is returned back to the calling routine (sub-program RATE2() or SOLVER()) and the sub-program RATE3() is ended.**

```
      Return
      End
```

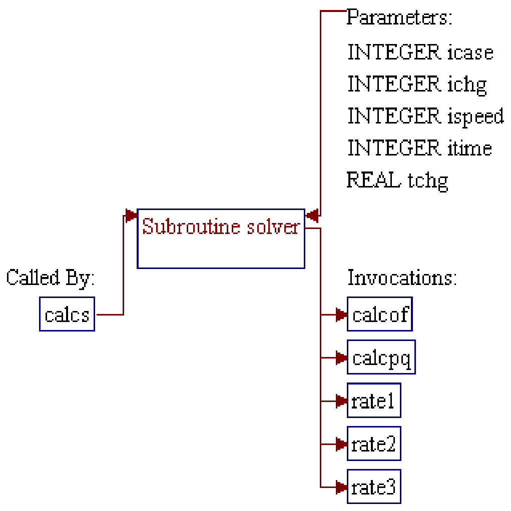
SUB-PROGRAM SOLVER()

LOCATION:	MODULE6C.FOR
MAIN THEME:	This routine solves for flow rates and pressures within specified time step.

CALLS:	CALCOF() (in file MODULE6B.FOR)
	Calculates open flow potentials.
	
	CALCPQ() (in file MODULE6D.FOR)
	Computes wellhead and bottom hole pressures after rates have been determined.
	
	RATE1() (in file MODULE6B.FOR)
	Calculates gas flow rates for primary wells under pressure constraint.
	
	RATE2() (in file MODULE6B.FOR)
	Calculates flow rates for primary wells based on rate constraint. Calculates each pay grade separately using same wellhead pressure.
	
	RATE3() (in file MODULE6B.FOR)
	Calculates gas flow rates for infill wells (infill once) under pressure constraint.

CALLED BY:	CALCS() (in file MODULE6D.FOR) Performs numerical convolution and solves for pressure and flow rates at each time step to generate type curves.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included. Common block and local variables are declared.**

Note: Name of the sub-program is SOLVER() and the parameters passed to this sub-program are as follows:

- *ITime* Time step number
- *ICase* Case number: 1=primary, 2=automatic refrac, 3=automatic infill one time, 4=automatic infill twice (not yet implemented).
- *IChg* Flag to indicate whether development type change has taken place: 0=primary wells only, 1=one set of infills, 2=two sets of infills (not currently used), 3= primary wells after restimulation
- *TChg* Time at which automatic change in development type occurs (automatic infill or refrac)
- *Ispeed* Flag for standard formula (*ispeed=0*, higher accuracy) or simplified formula (*ispeed=1*, lower accuracy, faster)

```
      SUBROUTINE Solver (ITime,  ICase,  IChg,   TChg,   ISpeed)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
```

Note: Additional common block and some local variables are declared.

```
      Common /stchg/iwin_yr
      real*4 qtini(qyr),pwhini(qyr),q1(3,qyr),q2(3,qyr),q3(3,qyr)
```

Step 2: **Flow rates for cases without infill wells are solved.**

```
         If ((IChg .eq. 0) .or. (IChg .eq. 3).or.tchg.lt.0.) then
```

Note: The following codes check flow rates based on minimum wellhead pressure (*PreMin*), no infilling. Sub-program RATE1() is invoked to calculate flow rate of primary wells under pressure constraint.

```
                 Pwh   = PreMin
                 IWell = 1
                 Call RATE1 (Pwh, QTotal, ITime, IChg, IWell)
                 if(icase.eq.1)then
                   qtini(itime)=qtotal
                   pwhini(itime)=pwh
                   do ipay=1,3
                    q1(ipay,itime)=qg(ipay,1,itime)
                    q2(ipay,itime)=qg(ipay,2,itime)
                    q3(ipay,itime)=qg(ipay,3,itime)
                   enddo
                  endif
                 If ((QTotal.lt.RatMax-1.).and.(IChg .eq. 0).and.
     +               (TChg.lt.0.))then
                     TChg = Time(ITime)
                     if(itime.lt.3)tchg=-1
                 endif
                      if(iwin_yr.eq.-1.and.tchg.gt.0.)then
                       If(itime.ge.3)then
                        iwin_yr=int(tchg)
                        if(Time(1).ne.1.0)
     &                  iwin_yr=int(tchg-Time(1))
                       endif
                      endif
```

Note: Check whether maximum flow rate (*RatMax*) is exceeded. If it is, sub-program RATE2() is invoked to calculate rate under flow rate constraint.

```
                 If (QTotal .ge. RatMax-1.0)then
                     Call RATE2 (Pwh,QTotal,ITime,IChg)
                 elseif(tchg.lt.0)then
                   qtotal=qtini(itime)
                   pwh=pwhini(itime)
                   do ipay=1,3
                    qg(ipay,1,itime)=q1(ipay,itime)
                    qg(ipay,2,itime)=q2(ipay,itime)
                    qg(ipay,3,itime)=q3(ipay,itime)
                   enddo
                   goto 997
```

Note: For cases with automatic infilling (*ICase=3* or *4*), add infill wells. Sub-program RATE3() is invoked to calculate flow rate of infill wells under pressure constraint.

```
                 Else If ((ICase .eq. 3) .or. (ICase .eq. 4)) then
                     IChg = 1
                     TChg = Time(ITime)
                     if(itime.lt.3)then
                      ichg=0
                      tchg=-1.
                     endif
                     if(itime.ge.3)then
                      Call RATE3 (PreMin,QTotal,ITime,IChg)
                      If (QTotal.gt.RatMax-1.)then
                       Call RATE2(Pwh,QTotal,ITime,ichg)
                      endif
                     else
                      qtotal=qtini(itime)
                      pwh=pwhini(itime)
                      do ipay=1,3
                       qg(ipay,1,itime)=q1(ipay,itime)
                       qg(ipay,2,itime)=q2(ipay,itime)
                       qg(ipay,3,itime)=q3(ipay,itime)
                      enddo
                     endif
```

Note: For cases with automatic refracturing (*ICase=2*) before stimulation (*IChg<>3*), refrac the wells. Sub-program RATE1() is invoked to calculate flow rate of primary wells under pressure constraint. If maximum flow rate (*RatMax*) is exceeded, sub-program RATE2() is invoked to calculate rate under flow rate constraint.

```
                Else If ((ICase .eq. 2).and. (IChg .ne. 3)) then
                    IChg  = 3
                    TChg  = Time(ITime)
                    IWell = 1
                    if(itime.lt.3)then
                     tchg=-1
                     ichg=0
                    endif
                    Call RATE1 (PreMin, QTotal, ITime, IChg, IWell)
                    If (QTotal .gt. RatMax-1.0.or.tchg.lt.0)then
                     ichg=0
                     iwell=1
                     Call RATE2(Pwh,QTotal,ITime,ichg)
                    endif
                End If
```

Step 3: **Flow rates for cases with infill wells are solved.**

Note: Sub-program RATE3() is invoked to calculate flow rate of infill wells under pressure constraint. If maximum flow rate (*RatMax*) is exceeded, sub-program RATE2() is invoked to calculate rate under flow rate constraint.

```
         Else
                Call RATE3 (PreMin,QTotal,ITime,IChg)
                If (QTotal.gt.RatMax-1.)
      %             Call RATE2(Pwh,QTotal,ITime,ichg)
         End If
```

Step 4: **Bottomhole pressure, wellhead pressure, and open flow potentials are determined.**

Note: Sub-program CALCPQ() is invoked to calculate bottomhole and wellhead pressures based on the calculated flow rates. Sub-program CALCOF() is invoked to calculate open flow potentials.

```
 997    Call CALCPQ (Pwh,   QTotal, ITime, IChg)
        Call CALCOF (ITime, ICase,  IChg,  ISpeed)
```

Step 5: **The program control is returned back to the calling routine (sub-program CALCS()) and the sub-program SOLVER() is ended.**

```
        Return
        End
```

SUB-PROGRAM WARREN()

LOCATION: MODULE6C.FOR

MAIN THEME: This routine calculates the difference in dimensionless pressures between a conventional reservoir and a naturally fractured reservoir using Warren and Root approach.

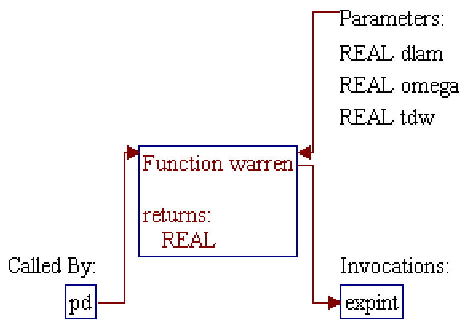
CALLS: EXPINT() (in file MODULE6D.FOR)
Computes exponential integral function.

CALLED BY: PD() (in file MODULE6B.FOR)
Calculates dimensionless pressure functions for different reservoir systems.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared.**

Note: Name of the sub-program is WARREN() and the parameters passed to this sub-program are as follows:

•	*Tdw*	Dimensionless time based on wellbore radius
•	*Omega*	Warren and Root porosity-compressibility ratio: $Omega = \phi c_{fracture} / \phi c_{total}$
•	*DLam*	Warren and Root interporosity flow parameter: $DLam = 12 * k_{matrix} / k_{total} * (Rw / FracSpacing)^2$
•	*Warren*	Computed dimensionless pressure difference

```
         FUNCTION Warren(Tdw, Omega, DLam)
```

Step 2: **Set dimensionless pressure difference (*Warren*) equals to zero if Warren and Root parameters are out of range.**

```
         If ((Omega.le.0.) .or. (Omega.ge.1.) .or. (DLam.le.0.)) then
             Warren = 0.
```

Step 3: **Calculate dimensionless pressure difference (*Warren*).**

Note: Arguments of the exponential integrals for dimensionless pressure determinations (*Arg1* and *Arg2*) are calculated. Sub-program EXPINT() is called twice to calculate dimensionless pressures *P1* and *P2*.

```
         Else
             Arg1   = DLam * Tdw / Omega / (1. - Omega)
             Arg2   = Arg1 * Omega
             P1     = ExpInt(Arg1)
             P2     = ExpInt(Arg2)
             Warren = 0.5 * (-P1 + P2)
         End If
```

Step 4: **Program control is returned back to the calling routine (sub-program PD()) and the sub-program WARREN() is ended.**

```
         Return
         End
```

SUB-PROGRAM WDRIVE()

LOCATION: MODULE6C.FOR

MAIN THEME: This routine computes performances of water drive reservoirs. Average reservoir pressure in each pay grade is directly calculated from water influx material balance. Each pay grade is assumed to act independently of the others. The design rate is allocated among the pay grades based on the original gas in place of each active pay grade. The pay grade is shut in when the net water influx (after water production is subtracted) is sufficient to fill the reservoir with water and gas trapped at the average pressure.

CALLS: CALCOF() (in file MODULE6B.FOR)
Calculates open flow potentials.

CALCPQ() (in file MODULE6D.FOR)
Computes wellhead and bottom hole pressures after rates have been determined.

CONVLV () (in file MODULE6C.FOR)
Performs numerical convolution to determine pressure drop caused by previous production.

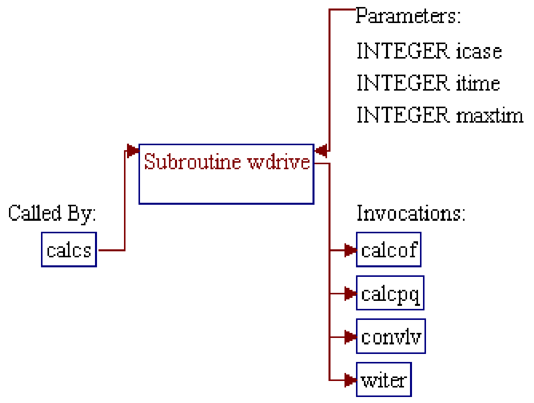
WITER() (in file MODULE6C.FOR)
Calculates water influx and based on that decides whether production from a specific water drive reservoir needs to be stopped (if water fills up the reservoir) or not.

CALLED BY: CALCS() (in file MODULE6D.FOR)
Performs numerical convolution and solves for pressure and flow rates at each time step to generate type curves.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included. Local variable is declared.**

Note: Name of the sub-program is WDRIVE() and the parameters passed to this sub-program are as follows:

• *ITime*	Time step number
• *ICase*	Case number: 1=primary, 2=automatic refrac, 3=automatic infill one time, 4=automatic infill twice (not yet implemented).
• *MAXTIM*	Maximum number of time steps (limited to 41 steps)

```
      SUBROUTINE WDrive (ITime,ICase,MaxTim)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'type2.h'
      include 'type5.h'
      include 'type3.h'
      include 'type4.h'
      include 'type8.h'
      include 'type7.h'
      include 'type6.h'
      include 'type10.h'
```

Note: Local variable for number of wells in pay grades (*Wells()*) is declared.

```
      Dimension Wells(3)
```

Step 2: **Number of wells (*Wells()*), total original gas in place *(G)*, and water influx (*WePref()*) are calculated/assigned.**

```
     G = 0.
      Do J = 1, 3
              Wells(J) = Area(J) / WSpace(J)
              G = G + OGIP1(J) * Wells(J)
              WePrev(J) = We(J)
     End Do
```

Step 3: **Occurrence of infill wells is checked.**

Note: Flag *IChg* (indicator for development type change) and factor *Div* (factor to divide or multiply flow rates: 1=no infill, 2=with infills) are assigned acordingly.

```
         If ((ICase.eq.3) .and. (Time(ITime).gt.(TimChg+0.0001))) then
                IChg  = 1
                Div   = 2.
         Else
                 IChg = 0
                Div = 1.
         End If
```

Step 4: Gas flow rates (*Qg()*) are calculated. Sub-program WITER() is invoked to calculate water influx and to decide whether the water has filled the reservoir.

Note: Flag of pay grade (*JSolv*) and pay grade loop are initialized.

```
         JSolv = 0
         Do J = 1, 3
```

Note: Time step size *(DT)* is set. Array variable *Time()* stores time data (e.g. 1, 2, 3,...) to be analyzed. The time step size is calculated by subtracting years from two consecutive data points. The size of the first time step is equal to the number of years in the first data point.

```
            DT  = Time(ITime)
            If (ITime.gt.1) DT  = Time(ITime)-Time(ITime-1)
```

Note: Gas production (*GP*) and gas production rate (*GasRat*) are calculated to be used in sub-program WITER().

```
            GpTot = RatMax*Time(ITime)*365.+
      +             RatMax*(Time(ITime)-TimChg)*365.*(Div-1.)
            Ratio    = OGIP1(J) / G
            GasRat   = RatMax * Ratio * Div
            Gp       = GpTot  * Ratio
            If (ITime.gt.1) then
                If (GasRat.gt. (Qg(J,1,ITime-1)*Div+1.)) then
                       GasRat = Qg(J,1,ITime-1)*Div
                       Gp     = CumGas(J,1,ITime-1)+CumGas(J,2,ITime-1)+
      +                         GasRat * DT * 365.
                 End If
            End If
```

Note: If the pay grade is on production (*KShut(J)=0*), sub-program WITER() is invoked to calculate water influx and to check whether the reservoir needs to be shut in or not (*IShut*: 0=OK, 1=shut in). The reservoir is shut in if the water fills up the reservoir.

```
            If (KShut(J).eq.0) then
                 Call WITER (ITime, MaxTim, J, Gp, GasRat, IShut)
```

Note: Gas production and rate are adjusted based on primary or infill wells (divided by *Div*).

```
                 Qg(J,1,ITime) = GasRat/Div
                 If (IChg .eq.1) Qg(J,2,ITime) = GasRat/Div
```

Step 5: The water fill-up time is determined.

Note: If water influx fills up the reservoir, the time when the water fill-up occurs is determined using 12 Bi-section iterations. First, minimum and maximum time (*T0* and *T1*) for Bi-section iteration are set.

```
                 If (IShut.eq.1) then
                     Maxtim = Min(Maxtim,41)
                     KMin = Max(2,ITime+1)
                     Do K = Maxtim, KMin, -1
                         Time(K) = Time(K-1)
                     End Do
                     T1 = Time(ITime)
                     T0 = 0.
                     If (ITime.gt.1) T0 = Time(ITime-1)
```

Note: Bi-section iteration is performed. In each iteration, the fill-up time (*TIme(ITime)*) is estimated as a middle point between *T0* and *T1*. Expected gas production (*GP*) and gas rate (*GasRat*) are calculated and passed to sub-program WITER(). The resulting shut-in or not decision (*IShut=1* or *0*) from sub-program WITER() is used to reduce the span between *T0* and *T1*. This process is repeated until the difference between *T0* and *T1* is at most 0.0001 years.

```
                     Do K = 1, 12
                         Time(ITime) = (T1 + T0) / 2.
                         DT  = Time(ITime)
                         If (ITime.gt.1) DT  = Time(ITime)-Time(ITime-1)
                         KShut(J) = 0
                         IShut    = 0
                         Ratio    = OGIP1(J) / G
                         GasRat   = RatMax * Ratio * Div
                         Gp   = RatMax*Time(ITime)*365.+
     +                          RatMax*(Time(ITime)-TimChg)*365.*(Div-1.)
                         If (ITime.gt.1) then
                             If (GasRat.gt.(Qg(J,1,ITime-1)*Div+1.)) then
                                 GasRat = Qg(J,1,ITime-1)*Div
                                 Gp = CumGas(J,1,ITime-1)
     +                              + CumGas(J,2,ITime-1)
     +                              + GasRat * DT * 365.
                             End If
                         End If
                         If (abs(T1-T0).gt.0.0001) then
                             Call WITER (ITime,MaxTim,J,Gp,GasRat,IShut)
                             If (IShut.eq.1) then
                                 T1 = Time(ITime)
                             Else
                                 T0 = Time(ITime)
                             End If
                         End If
                     End Do
                     KShut(J) = 1
                     JSolv    = J
                 End If
            End If
         End Do
```

Step 6: **Flow rates from other pay grade reservoirs are re-calculated if one of the pay grade is shut in where the calculation is based on the corrected time step.**

```
         Do J = 1, 3
            DT       = Time(ITime)
            If (ITime.gt.1) DT  = Time(ITime)-Time(ITime-1)
            Ratio    = OGIP1(J) / G
            GasRat   = RatMax * Ratio * Div
            GpTot    = RatMax*Time(ITime)*365.+
     +                 RatMax*(Time(ITime)-TimChg)*365.*(Div-1.)
            Gp       = GpTot  * Ratio
            If (ITime.gt.1) then
               If (GasRat.gt.(Qg(J,1,ITime-1)*Div+1.)) then
                 GasRat = Qg(J,1,ITime-1)*Div
                 Gp = CumGas(J,1,ITime-1)+CumGas(J,2,ITime-1)+
     +                  GasRat*DT*365.
               End If
            End If
            If ((KShut(J).eq.0).and.(JSolv.gt.J)) then
                 Call WITER (ITime, MaxTim, J, Gp, GasRat, IShut)
                 Qg(J,1,ITime) = GasRat/Div
                 If (IChg .eq.1) Qg(J,2,ITime) = GasRat/Div
            End If
         End Do
```

Step 7: **Sub-program CONVLV() is invoked to calculate pressure drops due to changes of flow rates in previous time step (superposition in time).**

```
         JConv = 2
         Call CONVLV (ITime, JConv, ISpeed)
```

Step 8: **Sub-program CALCPQ() is invoked to calculate wellhead and bottomhole pressures.**

```
         Call CALCPQ (Pwh,   QTotal, ITime, IChg)
```

Step 9: **Flow rates are reduced if the reservoir cannot sustain the desired flow rates.**

Note: The calculated and minimum allowed wellhead pressures (*Prwh()* and *PreMin*) are compared. If *Prwh()* is lower than *PreMin* a 12-Bi-section iteration is performed to correct the flow rates. Minimum and maximum flow rates (*Q0* and *Q1*) for the Bi-section iteration are taken as zero and the total of flow rates from primary and infill wells. A flow rate tolerance of 0.1 MCFD is used as a convergence criterion.

```
Do J = 1, 3
     If (Prwh(J,1,ITime).lt.PreMin) then
        DT  = Time(ITime)
        Gp0 = 0
        If (ITime.gt.1) then
                DT  = Time(ITime)-Time(ITime-1)
                Gp0 = CumGas(J,1,ITime-1) + CumGas(J,2,ITime-1)
        End If
        Q1 = Qg(J,1,ITime)+Qg(J,2,ITime)
        Q0 = 0
        Do K = 1, 12
          If (abs(Q1-Q0).gt.0.1) then
            GasRat = (Q1+Q0)/2.
            Gp = Gp0 + GasRat * DT * 365.
            Call WITER (ITime, MaxTim, J, Gp, GasRat, IShut)
            Qg(J,1,ITime) = GasRat/Div
            If (IChg.eq.1) Qg(J,2,ITime) = GasRat/Div
            JConv = 2
            Call CONVLV (ITime, JConv, ISpeed)
            Call CALCPQ (Pwh,   QTotal, ITime, IChg)
            If (Prwh(J,1,ITime).lt.PreMin) then
                Q1 = GasRat
            Else
                Q0 = GasRat
            End If
          End If
        End Do
    End If
End Do
```

Step 10: **Sub-programs CONVLV() and CALCPQ() are re-invoked to determine the wellhead and bottomhole pressures based on the corrected flow rates.**

```
Call CALCPQ (Pwh,   QTotal, ITime, IChg)
Call CALCOF (ITime, ICase,  IChg,  ISpeed)
```

Step 11: **The program control is returned back to the calling routine (sub-program CALCS()), and the sub-program WDRIVE() is ended.**

```
Return
End
```

SUB-PROGRAM WET()

LOCATION: MODULE6A.FOR

MAIN THEME: This routine is a type curve module for wet coal and wet shale reservoirs that drives sub-routine WETQ() to calculate gas flow rates. The module implements material balance directly to compute average reservoir pressure in each pay grade.

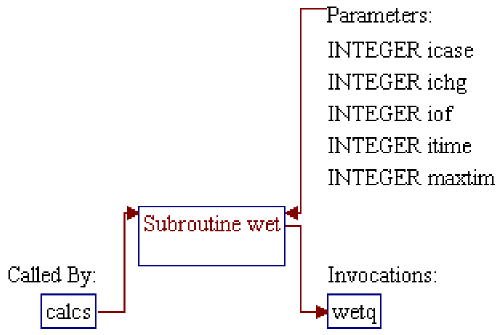
CALLS: WETQ() (in file MODULE6A.FOR)
Calculates gas flow rates for wet coal and wet shale reservoirs based on bottom hole pressure.

CALLED BY: CALCS() (in file MODULE6D.FOR)
Performs numerical convolution and solves for pressure and flow rates at each time step to generate type curves.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included.**

Note: Name of the sub-program is WET() and the parameters passed to this sub-program are as follows:

•	*ITime*	Time step number
•	*ICase*	Case number: 1=primary, 2=automatic refrac, 3=automatic infill one time.
•	*IChg*	Flag to indicate whether development type change has taken place: 0=primary wells only, 1=one set of infills, 2=two sets of infills (not currently used), 3= primary wells after restimulation
•	*MaxTim*	Maximum number of time steps
•	*IOF*	Flag to indicate type of calculation: 0=normal, 1=open flow calculation

```
      SUBROUTINE WET (ITime,ICase,IChg,MaxTim,IOF)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'type2.h'
      include 'type5.h'
      include 'type3.h'
      include 'type4.h'
      include 'type8.h'
      include 'type7.h'
      include 'type6.h'
      include 'type10.h'
```

Step 2: **Amount of gas adsorbed at current time level is calculated.**

Note: Pay grade loop and variables are initialized. Bulk volume (*VB*) is calculated.

```
      Do I = 1, 3
          Sum = 0.
          B0  = 0.
          F0  = 0.
          VB  = WSpace(I) * Thick(I)
```

Note: Time step loop, previous time level (*T*), initial pressure (*Pi*), and average reservoir pressure (*P*) are initialized.

```
          Do J = 1, ITime
              T = 0.
              If (J.gt.1) T = Time (J-1)
              Pi = Pinit(I)
              P  = PreAvg(I,J)
```

Note: Equilibrium amount of gas adsorbed (*F*), steady-state sorption rate (*B*), time step to sorption time constant ratio (*DT*), and factor *E* (exponential of *DT*) at time step *J* are calculated. This module implements dual-porosity approach (non-equilibrium sorption approach) by taking into consideration sorption time constant (*TDes()*).

```
              F  = VL(I) * (Pi/(Pi+PL(I)) - P/(P+PL(I)))
              B  = (F-F0)/(Time(J)-T)
              DT = (Time(ITime)-T)/(TDes(I)/365.)
              E  = 0.
              If (DT.lt.30.) E = exp(-DT)
```

Note: Amount of gas adsorbed (*ADesrb()*) is calculated using non-equilibrium formulation.

```
              If (J.ne.ITime) then
                  Sum = Sum + E * (B-B0)
              Else
                  ADesrb(I) =(Sum+F0/DT*(1.-E)-E*B0)*VB*RhoMa(I)/1000.
                  BDesrb(I) = (1. - (1.-E)/DT) * VB * RhoMa(I) / 1000.
                  ADesrb(I) = ADesrb(I)+BDesrb(I)*VL(I)*Pi/(Pi+PL(I))
              End If
```

Note: Values of *B* and *F* are stored for calculation at the next time step. The time step and pay grade loops are closed.

```
              B0 = B
              F0 = F
          End Do
      End Do
```

Step 3: **Value for *IChg* is set to one if infills are present or it is set to zero otherwise.**

```
      If ((ICase.eq.3) .and. (Time(ITime).gt.(TimChg+0.0001))) then
              IChg  = 1
      Else
              IChg = 0
      End If
```

Step 4: **Gas flow rates based on absolute open flow (*CAOF()*) are calculated.**

Note: For absolute open flow calculation, bottom hole pressure (*Pbh*) is set to atmospheric pressure (14.7 psia). Since this calculation does not limit the flow rates, the maximum flow rates (*QwMax()*) are temporarily set to a big number (1E6). Sub-program WETQ() is then invoked to calculate the gas flow rates and the results are stored to variable *CAOF()*.

```
Pbh = 14.7
T = Time(ITime)
Time(ITime) = 1.
R = RatMax
RatMax = 1000000.
Qw1 = QwMax(1)
Qw2 = QwMax(2)
Qw3 = QwMax(3)
QwMax(1) = 1000000.
QwMax(2) = 1000000.
QwMax(3) = 1000000.
If (ITime .gt. 1) Time(ITime) = Time(ITime-1) + 1.
Call WetQ (Pbh, QTotal, ITime, IChg)
Time(ITime) = T
RatMax = R
QwMax(1) = Qw1
QwMax(2) = Qw2
QwMax(3) = Qw3
Do J = 1, 3
   Do K = 1, 3
        CAOF(J,K,ITime) = Qg(J,K,ITime)
   End Do
End Do
```

Step 5: **Bottom hole pressure is set to a minimum allowable wellhead pressure (*PreMin*) and sub-program WETQ() is invoked to calculate the maximum total gas flow rate (*QTotal*).**

```
Pbh = PreMin
Call WetQ (Pbh, QTotal, ITime, IChg)
```

Step 6: **Bottom hole pressure (*Pbh*) is re-calculated if the maximum total gas flowa rate (*QTotal*) is higher than the user specified maximum gas rate (*RatMax*).**

Note: The following codes calculate *Pbh* iteratively until the calculated *QTotal* (from sub-program WETQ()) is close to the specified *RatMax*. First, a 5-Bi-section-iteration is performed to get good estimation of *Pbh* for Newton-Raphson iteration.

```
If (QTotal.gt.RatMax) then
        P0 = Pbh
        P1 = Max(Pinit(1),Pinit(2),Pinit(3))
        If (ITime.gt.1) P1 = Max(PreAvg(1,ITime-1),
     +                          PreAvg(2,ITime-1),PreAvg(3,ITime-1))
                 Do I = 1, 5
                    Pbh = (P1+P0) / 2.
                    Call WetQ (Pbh, QTotal, ITime, IChg)
                    If (QTotal.gt.RatMax) then
                        P0 = Pbh
                    Else
                        P1 = Pbh
                    End If
                 End Do
```

Note: The calculation is continued with a maximum of 15 Newton-Raphson iterations.

```
                 Pbh = (P1+P0) / 2.
                 JSolv = 0
                 DP = 1.
                 Do I = 1, 15
                   If (JSolv.eq.0) then
                     Pbh = Pbh + DP
                     Call WetQ (Pbh, QTotal, ITime, IChg)
                     F1 = QTotal - RatMax
                     Pbh = Pbh - DP
                     Call WetQ (Pbh, QTotal, ITime, IChg)
                     F  = QTotal - RatMax
                     If (abs(F).le.0.1) then
                         JSolv = 1
                     Else
                         If (abs(F1-F) .lt. 0.000001) then
                            DP = DP * 2.
                         Else
                            DelP =  - F * DP / (F1 - F)
                            If ((Pbh+DelP) .le. P0) then
                               Pbh = P0 + 0.25 * (Pbh-P0)
                            Else If ((Pbh+DelP) .ge. P1) then
                               Pbh = P1 - 0.25 * (P1-Pbh)
                            Else
                               Pbh = Pbh + DelP
                            End If
                         End If
                     End If
                   End If
                 End Do
         End If
```

Step 7: **The program control is returned back to the calling routine (sub-program CALCS()) and the sub-program WET() is ended.**

```
         Return
         End
```

SUB-PROGRAM WETQ()

LOCATION: MODULE6A.FOR

MAIN THEME: This routine calculates gas flow rates for wet coal or shale reservoirs. The routine solves for flow rates of the primary well case (no infilling) under pressure constraint. The minimum flow rate constraint is zero and the maximum is based on the sandface pressure from previous withdrawals.

Due to uncertainty in producing fluid levels, pump efficiencies, etc., it has been assumed that the sandface pressure is specified for this case.

CALLS: CWATER() (in file MODULE6D.FOR)
Calculates water compressibility using OSIF's correlation (SPE Reservoir Engineering, Feb. 1988).

PSI() (in file MODULE6B.FOR)
Performs table look-up of real gas potential (psudo-pressure) given pressure.

PWELL() (in file MODULE6B.FOR)
Calculates bottomhole pressure, wellhead pressure, or flow rate based on the difference between well head pressure and bottom hole pressure of the well using Smith's formula.

RELPRM() (in file MODULE6A.FOR)
Calculates gas and water relative permeabilities for wet coal and shale reservoirs.

VISW() (in file MODULE6C.FOR)
Calculates water viscosity using Meehan's correlation.

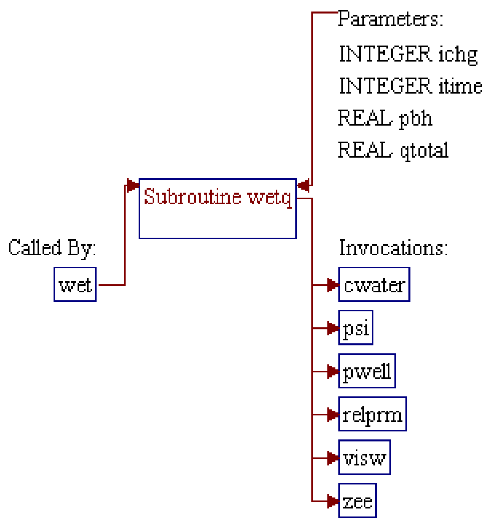
ZEE() (in file MODULE6C.FOR)
Performs table look-up linear interpolation of Z-factor as a function of pressure.

CALLED BY: WET() (in file MODULE6A.FOR)
Computes performances of wet coal and shale reservoirs using material balance approach.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included.**

Note: Name of the sub-program is WETQ() and the parameters passed to this sub-program are as follows:

•	*Pbh*	Bottom hole (sandface) pressure (psia)
•	*QTotal*	Total gas production from field (MCF/D)
•	*ITime*	Time step number
•	*IChg*	Flag to indicate whether development type change has taken place: 0=primary wells only, 1=one set of infills, 2=two sets of infills (not currently used), 3= primary wells after restimulation

```
         SUBROUTINE WetQ (Pbh, QTotal, ITime, IChg)
```

Note: Header .h files which declare global variables and common blocks are included.

```
         include'dimen.h'
         include'type1.h'
         include'type2.h'
         include'type3.h'
         include'type4.h'
         include'type5.h'
         include'type6.h'
         include'type7.h'
         include'type8.h'
         include'type9.h'
         include'type10.h'
```

Step 2: **Time step size *DT* is set.**

Note: Array variable *Time()* stores time data (e.g. 1, 2, 3,...) to be analyzed. The time step size is calculated by subtracting years from two consecutive data points. The size of the first time step is equal to the number of years in the first data point.

```
         DT = Time(ITime)
         If (ITime.gt.1) DT = DT - Time(ITime-1)
```

Step 3: **Loop for pay grades (loop *J*) is initialized.**

```
         Do J = 1, 3
```

Step 4: **Gas and water production rates, *Qg()* and *Qw()*, are set to zeros if the water influx has filled up the reservoir.**

Note: The gas and water production rates use three-parameter array variables *Qg(p1,p2,p3)* and *Qw(p1,p2,p3),* where *p1* is for pay grades (1,2,3), *p2* is for primary or infill (1=primary, 2=infill), and *p3* is for time step (1,2,3,...). These variables are set to zeros if the water influx has filled up the reservoir. This assignment is controlled by flag *Kshut()* which is an indicator to denote whether the reservoir has not yet filled with water (*Kshut()=0*) or the water influx has filled more than 80% of the reservoir (*Kshut()>0*).

```
      If (KShut(J).gt.0) then
          Qg(J,1,ITime) = 0.
          Qw(J,1,ITime) = 0.
          Qg(J,2,ITime) = 0.
          Qw(J,2,ITime) = 0.
```

Step 5: **The gas and water flow rates calculations are performed if the water saturation in the reservoir is still less than 80% (*KShut()=0*) . The first step is obtaining information on skin factor (*S*), effective wellbore radius (*Rweff*), and the horizontal well length or fracture half-length (*DLen*).**

Note: The skin factor is obtained from three-parameter array variable *Skin(p1,p2,p3)* where:

•	*p1*	Parameter for pay grade (1,2,3)
•	*p2*	Parameter for primary or infill wells (1=primary, 2=infill)
•	*p3*	Parameter for stimulation type (1=no refrac, 2=with refrac)

Note that parameters *p2* and *p3* are dependent on each other because the current version of the RP Module does not consider infilling with refracturing as a development case. Therefore, for *p2=2*, the *p3* can only have a value of *1*. The effective well bore radius and the horizontal well length/fracture half-length are obtained from variables *Rw()*, *HalfLn()*, respectively.

```
          S = Skin(J,1,1)
          If (IChg.eq.3) S = Skin(J,1,2)
          Rweff  = Rw  (J,1)
          DLen   = HalfLn(J,1)
```

Step 6: **Effective wellbore radius for fractured or horizontal well is recalculated.**

Note: The value of *DLen* of greater than *1* indicates either the system is fractured or the well is horizontal. In this case, the dimensionless fracture conductivity (*Fcd*) is calculated, and if the value is negative (for infinite conductivity fracture) the *Fcd* is set to 100000.

```
          If (DLen.gt.1.) then
              Fcd   = Cond(J,1) / (Perm(J)*DLen)
              If (Fcd .le. 0.) Fcd = 100000.
```

Note: The following codes calculate the effective wellbore radius for two different wells: fractured vertical well (*JTyp()=0*), and horizontal well without fracture (*JTyp()=1*).

```
              SHor  = 0.
              Rweff = DLen/2.
              If (JTyp(J,1) .eq. 1) then
                  DLen  = HorLen(J,1)
                  Ratio = Sqrt(PermV(J)/Perm(J))
                  Rwd   = Rw(J,1)/Thick(J)*Ratio
                  SHor  = -2.*Thick(J)/DLen/Ratio*
     +                        Log(2.*Asin(3.1415926*Rwd))
                  Rweff = DLen / 4. * Exp (-SHor)
              Else
                  Rweff = Rweff / (1. + 1.71 / Fcd)
              End If
              If (Rweff .lt. Rw(J,1)) Rweff = Rw(J,1)
          End If
```

Step 7: **Constant terms of productivity index for gas and water (*QconG* and *QConW*) are calculated.**

Note: These terms are calculated using a standard well bore equation. For infill wells (*IChg=1*), drainage area (*Re*) is assumed to be equally split with equal skin factors and the productivity indexes are multiplied by *2* (for two wells). In this step, water compressibility (*Cw*) and pore volume compressibility (*Cp*) are also determined to be utilized later in material balance calculations. The *Cp* for coal and shale reservoirs is assumed to be 0.00005/psi.

```
          Re     = Sqrt(43560.*WSpace(J)/3.1415926)
          PCon   = Max(Log(Re/Rweff) - 0.75 + S, 1.)
          QConG  = Perm(J)*Thick(J) / (1422.*(Tem+460.)*PCon)
          P      = Pinit(J)
          WtrVis = VisW  (P, Tem, Salin(J))
          Cw     = Cwater(P, Tem, Salin(J))
          Cp     = .00005
          QConW  = Perm(J)*Thick(J) / (141.2 * WtrVis * PCon)
```

```
                  If (IChg.eq.1) then
                      Re    = Re / Sqrt(2.)
                      PCon  = Max(Log(Re/Rweff) - 0.75 + S, 1.)
                      QConG = 2.*Perm(J)*Thick(J)/(1422.*(Tem+460.)*PCon)
                      QConW = 2.*Perm(J)*Thick(J) / (141.2*WtrVis * PCon)
                  End If
```

Step 8: **Cumulative gas and water productions are calculated.**

Note: Sub-program ZEE() (located in fie MODULE6C.FOR) is invoked to determine gas Z-factor at initial pressure (*Pinit()*).

```
                  Pi   = Pinit(J)
                  Zi   = Zee(Pi,NArray,PreAry,ZAry)
```

Note: Depth of the reservoir (*Depth1()*) is assigned to working variable *Dep* to be used later in sub-program PWELL(). Volume of gas at current time (*G1i*) is calculated.

```
                  Dep  = Depth1(J)
                  G1i  = 43560. * WSpace(J) * Thick(J) * Poros(J) *
     +                  (1.-Swi(J)) * 520./(Tem+460.)/ 14.7 /1000.*Pi/Zi
```

Note: Cumulative gas and water productions up to the last time step (*Gp0* and *Wp0*) are calculated. This calculation utilizes three-parameter arrays of cumulative gas and water productions *CumGas(p1,p2,p3)* and *Qw(p1,p2,p3)*, where:

- *p1* — Parameter for pay grade (1,2,3)
- *p2* — Parameter for primary or infill wells (1=primary, 2=infill once, 3=infill twice)
- *p3* — Parameter for time steps (1,2,...)

In this calculation, the time step variable *DT* is temporarily used. Its value is set back to the original by recalculating the time step as mentioned in Step 2.

```
                  Gp0  = 0.
                  Wp0  = 0.
                  If (ITime.gt.1) then
                     Gp0 = CumGas(J,1,ITime-1) +
     +                     CumGas(J,2,ITime-1) + 2.*CumGas(J,3,ITime-1)
                     Do I = 1, ITime – 1
                         DT = Time(I)
                         If (I.gt.1) DT = Time(I) – Time(I-1)
                          Wp0 = Wp0 + (Qw(J,1,I)+Qw(J,2,I)) * DT * 365.
                     End Do
                  End If
                  DT = Time(ITime)
                  If (ITime.gt.1) DT = DT - Time(ITime-1)
```

Step 9: **Data for the first part of iterative procedure (Bi-section method) to calculate average reservoir pressure (*P*) are prepared.**

Note: Set minimum pressure (*P0*) and maximum pressure (*P1*) for Bi-Section iteration. Bottom hole pressure (*Pbh*) is used as the minimum pressure, and the previous average reservoir pressure (*PreAvg(J,Itime-1)*) is used as the maximum pressure. Average pressure for the current time step will be between the minimum and maximum pressures (*P0* < *P* < *P1*).

```
P0    = Pbh
P1   = Pinit(J)
If (ITime .gt. 1) P1 = PreAvg(J,ITime-1)
```

Note: Get type of unconventional reservoir (*KunCon()*) and store the value to working variable *KTyp*. *KunCon()* can have a value of:

•	*0*	Dry coal reservoir
•	*1*	Wet coal reservoir
•	*2*	Dry shale reservoir
•	*3*	Wet shale reservoir

Choose which relative permeability curves to be used. Code for location or type of the coal/shale reservoir (*ITYP*) is as follows:

•	*1*	Eastern coal
•	*2*	Western coal
•	*3*	Antrim shale

```
KTyp = KUnCon(J)
If (KUnCon(J) .eq. 3) then
       ITyp = 3
Else If (ILoc(J) .le. 1) then
       ITyp = 1
Else
       ITyp = 2
End If
```

Step 10: **The first part of iterative procedure to calculate average reservoir pressure (*P*) is performed using 5 Bi-section iterations.**

Note: Iteration loop (*I*) is initialized. The average reservoir pressure (*P*) at current Bi-section iteration level is taken as the middle point between the minimum and maximum pressures.

```
         Do I = 1, 5
             P  = (P0 + P1) / 2.
```

Note: Set minimum and maximum water saturations (*Sw0* and *Sw1*) of Bi-section iteration for water saturation determination.

```
             Sw0   = 0.
             Vp    = 43560. * WSpace(J) * Thick(J) * Poros(J)
             SwCon = 5.6146 /( (1.-Cp*(Pi-P)) * Vp)
             Swa   = Swi(J) * (1.+Cw*(Pi-P)) / (1.-Cp*(Pi-P))
             Sw1   = Swa - SwCon * Wp0
             If (Sw1.gt.   1.) Sw1 =    1.
             If (Sw1.lt.0.001) Sw1 = 0.001
```

Note: Perform 20 Bi-section iterations to get water saturation (*Sw*) at average reservoir pressure (*P*). This calculation is based on water material balance where water relative permeability (for two-phase water-gas system) as a function of *Sw* are determined using sub-program RELPERM() which is located in file MODULE6A.FOR. Note that the water saturation iteration is nested in the average reservoir pressure iteration.

```
             Do ISw = 1, 20
                 Sw  = (Sw1+Sw0)/2.
                 Call RELPRM (Sw,ITyp,DKrg,DKrw)
                 Q2w = QConW * (P - Pbh) * DKrw
                 SwTest = Swa - SwCon * (Wp0+Q2w*DT*365./2.)
                 If (Sw.gt.SwTest) then
                         Sw1 = Sw
                 Else
                         Sw0 = Sw
                 End If
             End Do
```

Note: If water rate (*Q2w*) exceeds maximum water rate (*QwMax()*), the water rate is reduced to the maximum water rate and the bottom hole pressure is adjusted based on the corrected water rate.

```
             Pbha  = Pbh
             If ((Q2w.gt.QwMax(J)) .and. (QwMax(J).gt.1.)) then
                 Q2w   = QwMax(J)
                 Sw = Swa - SwCon * (Wp0+Q2w*DT*365./2.)
                 Call RELPRM (Sw,ITyp,DKrg,DKrw)
                 Pbha  = P - Q2w/(QConW * DKrw)
             End If
```

Note: If water rate (*Q2w*) exceeds maximum water rate (*QwMax()*), the water rate is reduced to the maximum water rate and the bottom hole pressure is adjusted based on the corrected water rate.

```
                     Pbha  = Pbh
                     If ((Q2w.gt.QwMax(J)) .and. (QwMax(J).gt.1.)) then
                         Q2w   = QwMax(J)
                         Sw = Swa - SwCon * (Wp0+Q2w*DT*365./2.)
                         Call RELPRM (Sw,ITyp,DKrg,DKrw)
                         Pbha  = P - Q2w/(QConW * DKrw)
                     End If
```

Note: Flow rate at current time step based on gas material balance equation (*Q1* or *Q*) is calculated. First the cumulative gas production at current time step (*Gp*) is calculated. Cumulative gas sorption for coal reservoir is added to the cumulative gas production equation by implementing Langmuir equation (*ADesrb(J) - BDesrb(J) * VL(J) * P/(P+PL(J))*). The gas flow rate is then calculated based on the current and previous time-level cumulative gas productions. If the change of development case has taken place (from primary to infill once, *IChg=1*), the well flow rate is divided by 2.

```
                     Z  = Zee(P,NArray,PreAry,ZAry)
                     Gp = G1i * (1.-(1.-Cp*(Pi-P))*P/Z/(Pi/Zi)) +
     +                    ADesrb(J) - BDesrb(J) * VL(J) * P/(P+PL(J))
                     Q1 = (Gp - Gp0) / (365.*DT)
                     Q  = Q1
                     If (IChg.eq.1) Q = Q1 / 2.
```

Note: Sub-program PSI() (located in file MODULE6B.FOR) is invoked twice to determine the average pseudo-pressure (*PsiAvg*) as a function of average pressure (*P*) and bottom-hole pseudo-pressure (*PsiBh*) as a function of bottom hole pressure (*Pbha*).

```
                     PsiAvg = Psi(P,NArray,PreAry,PsiAry)
                     PsiBh  = Psi(Pbha,NArray,PreAry,PsiAry)
```

Note: Flow rate at current time step based on the well bore equation (*Q2*) is calculated. The term *QConG*DKrg* in the *Q2* equation is the productivity index of the gas.

```
                     Q2     = QConG * DKrg * (PsiAvg - PsiBh)
```

Note: The minimum or maximum pressures (*P0* or *P1*) is updated based on the difference between *Q1* and *Q2* and the Bi-section iterative process is repeated until the iteration counter (*I*) equals to 5. At the end of this iteration, the *P*, *P0*, and *P1* are expected to be relatively close to each other so that the iteration can be continued using the Newton-Raphson iterative procedure which requires a good initial estimation of *P*.

```
                 If (Q1.gt.Q2) then
                     P0 = P
                 Else
                     P1 = P
                 End If
             End Do
```

Step 11: **The second part of the iterative procedure to calculate average reservoir pressure (*P*) is performed using Newton-Raphson procedure up to 15 iterations.**

Note: The Newton-Raphson iterative procedure is initialized. Initial guess of pressure (*P*) is taken as the middle point between pressures *P0* and *P1* from the Bi-section iteration. Epsilon pressure for numerical derivative (*DP*) is set equal to *1* psi and convergence flag (*JSolv*) is set to zero to indicate non-convergence condition.

```
             P  = (P0 + P1) / 2.
             DP = 1.
             JSolv = 0
```

Note: Iteration loop (*I*) is initialized. The iteration process is repeated if the results do not meet the convergence criterion.

```
             Do I = 1, 15
               If (JSolv .eq. 0) then
```

Note: The following codes calculate water saturation (*Sw*), flow rate based on material balance (*Q1*), and flow rate based on well bore equation (*Q2*) using the same procedure as in the Bi-section iteration evaluated at reservoir pressure of *P+DP*. The difference between *Q1* and *Q2* is stored in variable *F1*.

```
                 P = P + DP
                 Sw0   = 0.
                 Vp    = 43560. * WSpace(J) * Thick(J) * Poros(J)
                 SwCon = 5.6146 /( (1.-Cp*(Pi-P)) * Vp)
                 Swa   = (1.+Cw*(Pi-P)) / (1.-Cp*(Pi-P))
                 Sw1   = Swa - SwCon * Wp0
                 If (Sw1.gt.   1.) Sw1 =    1.
                 If (Sw1.lt.0.001) Sw1 = 0.001
                 Do ISw = 1, 20
                     Sw  = (Sw1+Sw0)/2.
                     Call RELPRM (Sw,ITyp,DKrg,DKrw)
                     Q2w = QConW * (P - Pbh) * DKrw
                     SwTest = Swa - SwCon * (Wp0+Q2w*DT*365./2.)

                     If (Sw.gt.SwTest) then
                             Sw1 = Sw
                     Else
                             Sw0 = Sw
```

```
                         End If
                     End Do
                     Pbha  = Pbh
                     If ((Q2w.gt.QwMax(J)) .and. (QwMax(J).gt.1.)) then
                         Q2w   = QwMax(J)
                         Sw = Swa - SwCon * (Wp0+Q2w*DT*365./2.)
                         Call RELPRM (Sw,ITyp,DKrg,DKrw)
                         Pbha  = P - Q2w/(QConW * DKrw)
                     End If
                     Z  = Zee(P,NArray,PreAry,ZAry)
                     Gp = G1i * (1.-(1.-Cp*(Pi-P))*P/Z/(Pi/Zi)) +
     +                    ADesrb(J) - BDesrb(J) * VL(J) * P/(P+PL(J))
                     Q1 = (Gp – Gp0) / (365.*DT)
                     Q  = Q1
                     If (IChg.eq.1) Q = Q1 / 2.
                     PsiAvg = Psi(P,NArray,PreAry,PsiAry)
                     PsiBh  = Psi(Pbha,NArray,PreAry,PsiAry)
                     Q2     = QConG * (PsiAvg - PsiBh) * DKrg
                     F1 = Q1 - Q2
```

Note: Now the reservoir pressure is set back to its original value and the *Sw*, *Q1*, and *Q2* are recalculated. The difference between *Q1* and *Q2* is stored at variable *F*.

```
                     P = P – DP
                     Sw0   = 0.
                     Vp    = 43560. * WSpace(J) * Thick(J) * Poros(J)
                     SwCon = 5.6146 /( (1.-Cp*(Pi-P)) * Vp)
                     Swa   = (1.+Cw*(Pi-P)) / (1.-Cp*(Pi-P))
                     Sw1   = Swa - SwCon * Wp0
                     If (Sw1.gt.   1.) Sw1 =    1.
                     If (Sw1.lt.0.001) Sw1 = 0.001
                     Do ISw = 1, 20
                         Sw  = (Sw1+Sw0)/2.
                         Call RELPRM (Sw,ITyp,DKrg,DKrw)
                         Q2w = QConW * (P - Pbh) * DKrw
                         SwTest = Swa - SwCon * (Wp0+Q2w*DT*365./2.)
                         If (Sw.gt.SwTest) then
                                 Sw1 = Sw
                         Else
                                 Sw0 = Sw
                         End If
                     End Do
                     Pbha  = Pbh
                     If ((Q2w.gt.QwMax(J)) .and. (QwMax(J).gt.1.)) then
                         Q2w   = QwMax(J)
                         Sw = Swa - SwCon * (Wp0+Q2w*DT*365./2.)
                         Call RELPRM (Sw,ITyp,DKrg,DKrw)
                         Pbha  = P - Q2w/(QConW * DKrw)
                     End If
                     Z  = Zee(P,NArray,PreAry,ZAry)
                     Gp = G1i * (1.-(1.-Cp*(Pi-P))*P/Z/(Pi/Zi)) +
     +                    ADesrb(J) - BDesrb(J) * VL(J) * P/(P+PL(J))
                     Q1 = (Gp - Gp0) / (365.*DT)
                     Q  = Q1
                     If (IChg.eq.1) Q = Q1 / 2.
                     PsiAvg = Psi(P,NArray,PreAry,PsiAry)
                     PsiBh  = Psi(Pbha,NArray,PreAry,PsiAry)
                     Q2     = QConG * (PsiAvg - PsiBh) * DKrg
                     F = Q1 - Q2
```

Note: The convergence is checked by evaluating the value of dependent variable (*F*). The convergence is achieved if the value of *F* is less than 0.1 MCF/well. Otherwise, improvement to the average reservoir pressure (*DelP*) is calculated, the pressure (*P*) is updated, and the iteration process is continued.

```
                     If (abs(F) .lt. 0.1) then
                         JSolv = 1
                     Else
                         If (abs(F1-F) .lt. 0.00001) then
                            DP = DP * 2.
                         Else
                            DelP = - F * DP / (F1 - F)
                            If ((P+DelP) .le. P0) then
                                P = P0 + 0.25 * (P-P0)
                            Else If ((P+DelP) .ge. P1) then
                                P = P1 - 0.25 * (P1-P)
                            Else
                                P = P + DelP
                            End If
                         End If
                     End If
                   End If
                 End Do
```

Step 12: **Average reservoir pressure, bottom hole pressure, and gas and water flow rates are stored to type curve variables.**

Note: The following codes store pressure and flow rates results to the type curve variables. Bottom hole pressure (*Prbh()*) and water flow rate (*Qw*) for the case with infill wells are updated accordingly.

```
                     PreAvg(J,ITime) = P
                     Qg(J,1,ITime)   = Q
                     Qw(J,1,ITime)   = Q2w
                     Prbh(J,1,ITime) = Pbha
                     Prbh(J,2,ITime) = Pbha + (1.+0.25/PCon)*(P-Pbha)
                     Prbh(J,3,ITime) = P
                     If (IChg.eq.1) then
                         Qg(J,2,ITime)   = Q
                         Qw(J,1,ITime)   = Q2w/2.
                         Qw(J,2,ITime)   = Q2w/2.
                         Prbh(J,2,ITime) = Pbha
                     End If
```

Step 13: **Sub-program PWELL() (located in file MODULE6B.FOR) is invoked to calculate wellhead pressure.**

Note: An integer “2” in the sixth parameter of the sub-program PWELL() tells the routine to calculate wellhead pressure given the bottom hole pressure.

```
                     Do K = 1, 3
                         Qk = Qg(J,K,ITime)
                         Pk = Prbh(J,K,ITime)
                         KTyp = KUnCon(J)
                         Call PWELL(Pwk,Pk,Qk,Deriv,Dep,2,
     +                              IErr,KTyp,J)
                         Prwh(J,K,ITime) = Pwk
                     End Do
```

Step 14: **Cumulative gas production is calculated, pay grade loop *(J)* is closed, and gas total flow rate is calculated.**

Note: After cumulative gas production calculation is performed, the pay grade loop (*J*) is closed. The gas total flow rate is then calculated for all pay grades.

```
                     If (ITime.eq.1) then
                         CumGas(J,1,ITime) = Qg(J,1,ITime) * DT * 365.
                         CumGas(J,2,ITime) = Qg(J,2,ITime) * DT * 365.
                     Else
                         CumGas(J,1,ITime) = CumGas(J,1,ITime-1)+
     +                                       Qg(J,1,ITime) * DT * 365.
                         CumGas(J,2,ITime) = CumGas(J,2,ITime-1)+
     +                                       Qg(J,2,ITime) * DT * 365.
                     End If
                 End If
             End Do
             QTotal = 0.
             Do I=1,3
                 QTotal = QTotal + Area(I) / WSpace(I) *
     +               (Qg(I,1,ITime) + Qg(I,2,ITime) + Qg(I,3,ITime)*2.)
             End Do
```

Step 15: **The program control is returned back to the calling routine (sub-program WET()) and the sub-program WETQ() is ended.**

```
             Return
             End
```

SUB-PROGRAM WITER()

LOCATION: MODULE6C.FOR

MAIN THEME: This routine calculates water influx and based on that decides whether production from a specific water drive reservoir needs to be stopped (if water fills up the reservoir) or not. The procedure uses water influx material balance directly to calculate average reservoir pressure in each pay grade. The pay grade is shut in when the net water influx (after water production is subtracted) is sufficient to fill the reservoir with water and gas trapped at the average pressure.

CALLS: CPOROS() (in file MODULE6D.FOR)
Computes pore volume compressibility using a curve fit to Hall's correlation.

CWATER() (in file MODULE6D.FOR)
Calculates water compressibility using OSIF's correlation (SPE Reservoir Engineering, Feb. 1988).

DIMWE() (in file MODULE6D.FOR)
Calculates dimensionless cumulative water influx based on tables presented by Van Everdingen and Hurst (1949).

VISW() (in file MODULE6C.FOR)
Calculates water viscosity using Meehan's correlation.

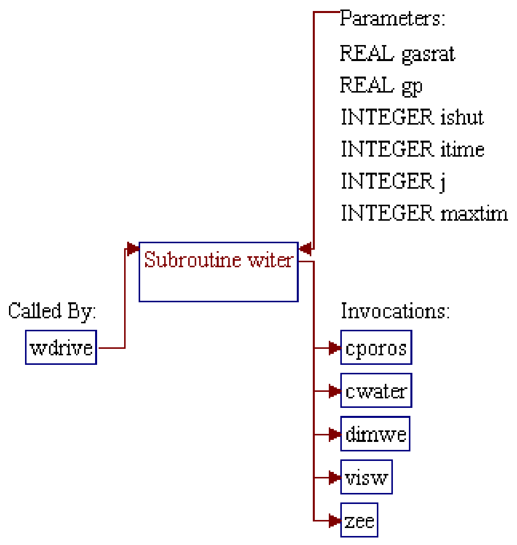
ZEE() (in file MODULE6C.FOR)
Performs table look-up linear interpolation of Z-factor as a function of pressure.

CALLED BY: WDRIVE() (in file MODULE6C.FOR)
Computes performances of water drive reservoirs using water influx material balance.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included.**

Note: Name of the sub-program is WITER() and the parameters passed to this sub-program are as follows:

• *ITime*	Time step number
• *ICase*	Case number: 1=primary, 2=automatic refrac, 3=automatic infill one time, 4=automatic infill twice (not yet implemented).
• *MAXTIM*	Maximum number of time steps (limited to 41 steps)
• *J*	Pay grade being evaluated
• *Gp*	Desired gas production (MCF)
• *GasRat*	Desired gas production rate (MCFD)
• *IShut*	Flag to denote whether pay grade is shut in (0=Not yet restricted, 1=Shut in ths time step)

```
      SUBROUTINE WIter (ITime, MaxTim, J, Gp, GasRat, IShut)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'type2.h'
      include 'type1.h'
      include 'type5.h'
      include 'type3.h'
      include 'type4.h'
      include 'type8.h'
      include 'type7.h'
      include 'type10.h'
```

Step 2: **Shut in condition is returned (*IShut=1*) if the reservoir is already shut in (*KShut()=1*).**

```
      If (KShut(J).eq.1) then
              IShut = 1
              Return
      End If
```

Step 3: **Water influx (*We()*) and water production (*Wp()*) up to current time step are calculated.**

Note: Water influx (*We()*), water production(*Wp()*), pressure (*P*), and shut-in flag (*IShut*) are initialized.

```
We(J) = 0.
Wp(J) = 0.
P     = Pinit(J)
If (ITime.gt.2) P = PreAvg(J,ITime-2)
IShut = 0
```

Note: Sub-programs CWATER(), CPOROS(), and VISW() are invoked to calculate water compressibility, pore volume compressibility, and water viscosity, respectively. Constants for dimensionless time (*TC*) and dimensionless flow rate (*QC*) calculations are calculated.

```
      Cw     =  Cwater(P, Tem, Salin(J))
      Cp     =  Cporos(Poros(J))
      Ctw    =  Cw + Cp
      WtrVis =  VisW (P, Tem, Salin(J))
      TC = 0.006328 * Perm(J) * 365. * 3.1415926 /
   +             (Poros(J)*WtrVis*Ctw*Area(J)*43560.)
      QC = Poros(J)*Ctw*Thick(J)*Area(J)*43560.
```

Note: Total water influx (*We()*) and total water production(*Wp()*) up to the current time step are calculated. The water influx calculation uses sub-program DIMWE() for dimensionless cumulative water influx. *Td* and *PCon* are dimensionless time and dimensionless flow rate, respectively.

```
Do I = 1, ITime
        If (I.eq.1) then
                DT = Time(ITime)
                DP = (Pinit(J)-PreAvg(J,I))/2.
        Else If (I.eq.2) then
                DT = Time(ITime)-Time(I-1)
                DP = (Pinit(J)-PreAvg(J,I))/2.
        Else
                DT = Time(ITime)-Time(I-1)
                DP = (PreAvg(J,I-2)-PreAvg(J,I))/2.
        End If
        Td = DT * TC
        If (I.ne.ITime) then
                DT = Time(I)
                If (I.gt.1) DT=Time(I)-Time(I-1)
                Wp(J) = Wp(J)+365.*DT*Qw(J,1,I)
                We(J) = We(J)+QC*DimWe(Td,KAqTyp(J))*DP
        Else
                PCon  = QC*DimWe(Td,KAqTyp(J))/2.
                We(J) = We(J) + PCon * P
        End If
End Do
```

Step 4: **The first part of Bi-section iteration on reservoir pressure (*P*) is performed.**

Note: The purpose of this iteration is to find a range of pressure (two pressures) from pressure table *PreAry()* to be used as minimum and maximum pressures in the second part of the Bi-section iteration. *PreAry()* is a table of pressure from zero to about 10% higher than the highest initial pressures. This table consists of *NArray* number of data (typically 99 points). In this procedure, a maximum of *Ln(NArray)/Ln(2)+1* number of iterations (about 7 iterations for *NArray* of 99 data points) is expected to be enough to get a range of pressure for the next part of the Bi-section iteration. The procedure is started by calculating gas Z-factor at initial pressure (*Zi* at *Pi*), assigning minimum and maximum indexes of pressure in the table (*N0* and *N1*), and calculating the maximum number of iterations (*NMax*).

```
         Pi    = Pinit(J)
         Zi    = Zee(Pi,NArray,PreAry,ZAry)
         NMax = Log(Float(NArray))/Log(2.) + 1
         N0 = 2
         N1 = NArray
```

Note: Iteration loop is initialized and convergence criterion is defined. The iteration process is stopped if the two pressures in the table designated by pointers *N0* and *N1* are next to each other.

```
         Do M = 1, NMax
                If ((N1-N0-1).gt.0) then
```

Note: Reservoir pressure is taken as a middle pressure between pressures at points *N0* and *N1*. Total water influx (*WeTot*) is then calculated based on this pressure.

```
                         N = (N1+N0)/2
                         P = PreAry(N)
                         WeTot = We(J) - PCon * P
```

Note: Total water production (*WpTot*) is calculated based on the total water influx (*WeTot*). *QwMax()* is a multi purpose parameter. It is a maximum flow rate per well (BPD/well) if its magnitude is greater than 1 (*QwMax()>1*), or a fraction of total influx it its magnitude is between zero and one (*0<QwMax()<1*), or it is a water to gas ratio (-BBL/MCF) if its magnitude is less than zero. Based on the value of *QwMax()*, the total water production (*WpTot*) is calculated accordingly.

```
                         If ((QwMax(J).gt.0.).and.(QwMax(J).lt.0.999))
      +                          then
```

```
                         WpTot = WeTot * QwMax(J)
                     Else
                         DT = Time(ITime)
                         If (ITime.gt.1) DT=Time(ITime)-Time(ITime-1)
                         WatRat = - GasRat * QwMax(J) * 5.6146
                         If (QwMax(J).gt.0.) WatRat=QwMax(J)*5.6146
                         DWp =Min((WatRat*DT*365.),(WeTot-WePrev(J)))
                         WpTot = Wp(J)*5.6146 + DWp
                     End If
```

Note: Total water production (*WpTot*) is calculated based on the total water influx (*WeTot*). *QwMax()* is a multi purpose parameter. It is a maximum flow rate per well (BPD/well) if its magnitude is greater than 1 (*QwMax()>1*), or a fraction of total influx it its magnitude is between zero and one (*0<QwMax()<1*), or it is a water to gas ratio (-BBL/MCF) if its magnitude is less than zero. Based on the value of *QwMax()*, the total water production (*WpTot*) is calculated accordingly. Once the total water influx and total water production are available, net water influx (*WeNet*) and dimensionless water influx (*WedNet*) can be calculated.

```
                     If ((QwMax(J).gt.0.).and.(QwMax(J).lt.0.999))
     +                       then
                         WpTot = WeTot * QwMax(J)
                     Else
                         DT = Time(ITime)
                         If (ITime.gt.1) DT=Time(ITime)-Time(ITime-1)
                         WatRat = - GasRat * QwMax(J) * 5.6146
                         If (QwMax(J).gt.0.) WatRat=QwMax(J)*5.6146
                         DWp =Min((WatRat*DT*365.),(WeTot-WePrev(J)))
                         WpTot = Wp(J)*5.6146 + DWp
                     End If
                     WeNet  = WeTot – WpTot
                     WedNet = WeNet/(Area(J)*43560.*Thick(J)*
     +                        Poros(J)*(1.-Swi(J)))
```

Note: Water influx material balance based on average reservoir pressure (*PAvg*) is performed. The material balance calculation will yield total amount of gas production (*GpCalc*). Based on the calculated gas production (*GpCalc*) and desired gas production (*Gp*), pointers for two pressures in the table are modified so that the distance between these pressures in the table is reduced. As the iteration process progresses, the *GpCalc* will be closer to *Gp*.

```
                     PAvg   = (Pi + P) / 2.
                     Cw     = Cwater(PAvg, Tem, Salin(J))
                     Cp     = Cporos(Poros(J))
                     Cwp    = (Cw * Swi(J) + Cp)/(1. - Swi(J))
                     GpCalc = OGIP1(J)*(1.-PreAry(N)/ZAry(N)/(Pi/Zi)*
     +                       (1. - Cwp * (Pi-P) -WedNet) )
                     If (GpCalc.lt.Gp) then
                             N1 = N
                     Else
                             N0 = N
                     End If
                 End If
             End Do
```

Step 5: **The second part of Bi-section iteration on reservoir pressure (*P*) is performed.**

Note: Similar procedure as in Step 4 is performed with different number of iteration, starting pressures, and convergence criterion. This time, the maximum number of Bi-section iteration is set to 12, the minimum and maximum pressures are the pressures obtained from Step 4, and the convergence criterion is based on the difference between reservoir pressures from two consecutive iteration level. The iteration process is repeated with pressure tolerance of 0.001 psi. At the end of the iteration, the difference between the calculated and desired gas productions is expected to be very small (converged).

```
      NMax = 12
      P0 = PreAry(N0)
      P1 = PreAry(N1)
      JSolv = 0
      Do M = 1, NMax
              If (JSolv.eq.0) then
                      P = (P0+P1) / 2.
                      Z = Zee(P,NArray,PreAry,ZAry)
                      WeTot = We(J) - PCon * P
                      If ((QwMax(J).gt.0.).and.(QwMax(J).lt.0.999))
   +                          then
                          WpTot = WeTot * QwMax(J)
                      Else
                          DT = Time(ITime)
                          If (ITime.gt.1) DT=Time(ITime)-Time(ITime-1)
                          WatRat = - GasRat * QwMax(J) * 5.6146
                          If (QwMax(J).gt.0.) WatRat=QwMax(J)*5.6146
                          DWp =Min((WatRat*DT*365.),(WeTot-WePrev(J)))
                          WpTot = Wp(J)*5.6146 + DWp
                      End If
                      WeNet  = WeTot – WpTot
                      WedNet = WeNet/(Area(J)*43560.*Thick(J)*
   +                           Poros(J)*(1.-Swi(J)))
                      PAvg   = (Pi + P) / 2.
                      Cw     = Cwater(PAvg, Tem, Salin(J))
                      Cp     = Cporos(Poros(J))
                      Cwp    = (Cw * Swi(J) + Cp)/(1. - Swi(J))
                      GpCalc = OGIP1(J)*(1.-(P/Z)/(Pi/Zi)*
   +                          (1. - Cwp * (Pi-P) -WedNet) )
                      If ((abs(P1-P0).lt.0.001).or.
   +                      (abs(Gp-GpCalc).lt.0.01)) JSolv = 1
                      If (GpCalc.lt.Gp) then
                              P1 = P
                      Else
                              P0 = P
                      End If
              End If
      End Do
```

Step 6: **Water influx parameters are stored, water rates and productions are calculated.**

Note: Average reservoir pressure (*PreAvg()*), dimensionless water influx (*WeD()*), time step size (*DT*), water production rates (*Qw()*), water influx (*We()* and *WtrInf()*), and water production rate (*Wp()*) are set/calculated.

```
      PreAvg(J,ITime) = P
      WeD(J) = WedNet
      DT = Time(ITime)
      If (ITime.gt.1) DT=Time(ITime)-Time(ITime-1)
      Qw(J,1,ITime) = Max((WpTot/5.6146-Wp(J))/DT/365.,0.)
      We(J) = WeTot
      Wp(J) = Wp(J) + Qw(J,1,ITime) * DT * 365.
      WtrInf(J,ITime) = We(J)/5.6146/1000.
```

Step 7: **Decision to continue or stop production is set.**

Note: Maximum possible water recovery with trapped gas (*WeMax*) is calculated. Net water influx (*WeNet*) is compared with *WeMax*. The reservoir is shut in (*IShut=1*) if the calculated net water influx is higher than the maximum possible water recovery. *SgTrap()* is a parameter for trapped gas saturation behind advancing water influx front (gas is assumed to be trapped at initial pressure).

```
      WeMax = Area(J)*43560.*Thick(J)*Poros(J)*
     +        ( 1. - Swi(J) -SgTrap(J) )
      If (WeNet.gt.WeMax) IShut = 1
```

Step 8: **The program control is returned back to the calling routine (sub-program WDRIVE()) and the sub-program WITER() is ended.**

```
      Return
      End
```

SUB-PROGRAM BW()

LOCATION: MODULE6B.FOR

MAIN THEME: This routine calculates water formation volume factor. A curve fit method based on Dodson and Standing is utilized for water saturated with natural gas.

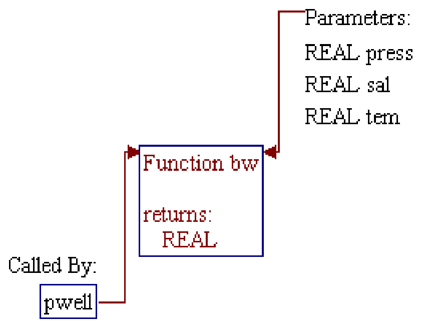
CALLS: None

CALLED BY: PWELL() (in file MODULE6B.FOR)
Calculates bottomhole pressure, wellhead pressure, or flow rate based on the difference between well head pressure and bottom hole pressure of the well using Smith's formula.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: Name and parameters of the sub-program are declared.

Note: Name of the sub-program is BW() and the parameters passed to this sub-program are as follows:

• *Press*	Pressure (psia)
• *Tem*	Temperature (degrees F)
• *Sal*	Water salinity (ppm by weight)
• *Bw*	Water formation volume factor (BBL/STB)

```
      FUNCTION   Bw (Press,  Tem,    Sal)
```

Step 2: Formation volume factor is calculated.

Note: In RP Module, the salinity of the brine is assumed to be zero (*Sal=0*).

```
      sal = 0.0
      Bw = 0.991663 - 1.465E-6 * Press + 5.984E-5 * Tem +
   +        8.48E-7 * Tem * Tem
```

Step 3: The program control is returned back to the calling routine (sub-program PWELL()) and the sub-program BW() is ended.

```
      Return
      End
```

SUB-PROGRAM CPOROS()

LOCATION: MODULE6D.FOR

MAIN THEME: This routine computes pore volume compressibility using a curve fit to Hall’s correlation.

CALLS: None

CALLED BY: CALCOF() (in file MODULE6B.FOR)
Calculates open flow potentials.

WITER() (in file MODULE6C.FOR)
Calculates water influx and based on that decides whether production from a specific water drive reservoir needs to be stopped (if water fills up the reservoir) or not.

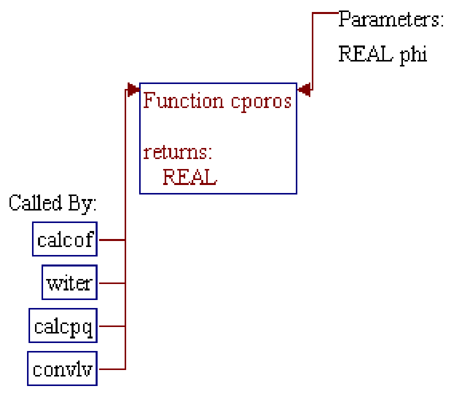
CALCPQ() (in file MODULE6D.FOR)
Computes wellhead and bottom hole pressures after rates have been determined.

CONVLV () (in file MODULE6C.FOR)
Performs numerical convolution to determine pressure drop caused by previous production.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared.**

Note: Name of the sub-program is CPOROS() and the parameters passed to this sub-program are as follows:

- *Phi* Porosity (fraction)
- *Cporos* Pore volume compressibility (1/psi)

```
      FUNCTION   Cporos (Phi)
```

Step 2: **Pore volume compressibility is calculated.**

Note: Two fitted equations are used for pore volume compressibility. The first one is for porosity less than 2% (*Phi<0.02*, tight formation) and the second equation for porosity higher or equal to 2%.

```
      If (Phi .lt. 0.02) then
              Cporos =  0.191976 / Phi * 1.0E-6
      Else
              Cporos = ((66.927 * Phi + 20.195 ) * Phi - 0.0735)
     +                 /(43.025 * Phi + 1.) / Phi * 1.0E-6
      End If
```

Step 3: **The program control is returned back to the calling routine (sub-program CALCOF(), WITER(), CALCPQ(), or CONVLV()) and the sub-program CPOROS() is ended.**

```
      Return
      End
```

SUB-PROGRAM CRIT()

LOCATION: MODULE6D.FOR

MAIN THEME: This routine computes pseudo-critical properties of natural gas using Standing's correlation with corrections for nitrogen, hydrogen sulfide, and carbon dioxide which are determined by the method of Wchert and Aziz (1974).

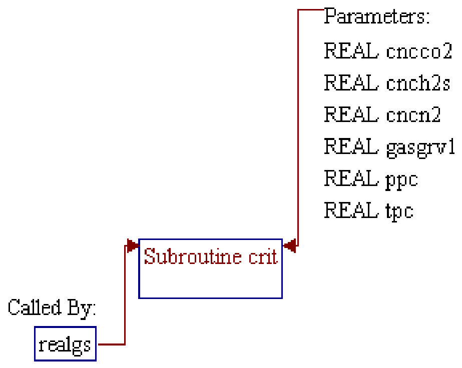
CALLS: None

CALLED BY: REALGS() (in file MODULE6C.FOR)
Calculates real gas potential (pseudo-pressure), gas viscosity, and gas compressibility factor (Z-factor) as functions of pressure.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared.**

Note: Name of the sub-program is CRIT() and the parameters passed to this sub-program are as follows:

Input Parameters:

•	*GasGrv1*	Natural gas specific gravity (air=1)
•	*CncH2S*	Concentration of hydrogen sulfide (fraction)
•	*CncCO2*	Concentration of carbon dioxide (fraction)
•	*CncN2*	Concentration of nitrogen (fraction)

Output Parameters:

•	*Tpc*	Pseudo-critical temperature (degree R)
•	*Ppc*	Pseudo-critical pressure (psia)

```
      SUBROUTINE Crit (GasGrv1, CncH2S, CncCO2, CncN2, Tpc, Ppc)
```

Step 2: **Properties of hydrocarbon fraction is calculated.**

```
      CncHC = (1. - CncN2 - CncCO2 - CncH2S)
      Ghc = (GasGrv1 - 0.967*CncN2 - 1.52*CncCO2 - 1.18*CncH2S)/CncHC
```

Step 3: **Specific gravity of hydrocarbon fraction is set at least equal to specific gravity of methane (0.554).**

```
      If (Ghc .lt. 0.554) Ghc = 0.554
```

Step 4: **Pseudo-critical temperature and pressure of the hydrocarbon fraction are calculated.**

```
      Ppchc = 677. + 15.0 * Ghc - 37.5 * Ghc * Ghc
      Tpchc = 168. + 325. * Ghc - 12.5 * Ghc * Ghc
```

Step 5: **Pseudo-critical temperature and pressure of the entire mixture is calculated.**

```
      Ppcm = Ppchc*CncHC + 493.*CncN2 + 1071.*CncCO2 + 1306.*CncH2S
      Tpcm = Tpchc*CncHC + 227.*CncN2 +  548.*CncCO2 +  672.*CncH2S
```

Step 6: **Wichert and Aziz correction to account for impurities is calculated and used to improve the value of pseudo-critical temperature and pressure.**

```
         Eps  = 120.*((CncH2S+CncCO2)**0.9 - (CncH2S+CncCO2)**1.6) +
      +       15.*(CncH2S**0.5-CncH2S**4.0)
         Tpc = Tpcm – Eps
         Ppc = Ppcm * Tpc / (Tpc + CncH2S * (1. - CncH2S) * Eps)
```

Step 7: **The program control is returned back to the calling routine (sub-program REALGS()) and the sub-program CRIT() is ended.**

```
         Return
         End
```

SUB-PROGRAM CWATER()

LOCATION: MODULE6D.FOR

MAIN THEME: This routine calculates water compressibility using Osif's correlation (SPE Reservoir Engineering, Feb. 1988, pp. 175-181). The nominal ranges for Osif's correlation are:

1000 < pressure < 2000 psia
0 < salinity < 200 gram/liter
200 < temperature < 270 degree F

Osif found that gas in solution with water had a small effect on compressibility that could be ignored for the correlation. The correlation generally gives accurate results even for temperature as low as 100 F.

CALLS: None

CALLED BY: WETQ() (in file MODULE6A.FOR)
Calculates gas flow rates for wet coal or shale reservoirs based on bottom hole pressure.

CALCOF() (in file MODULE6B.FOR)
Calculates open flow potentials.

WITER() (in file MODULE6C.FOR)
Calculates water influx and based on that decides whether production from a specific water drive reservoir needs to be stopped (if water fills up the reservoir) or not.

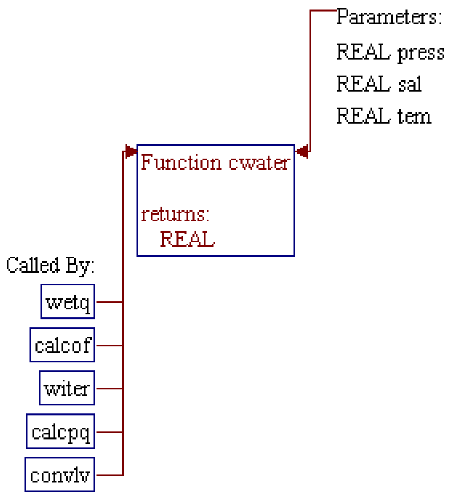
CALCPQ() (in file MODULE6D.FOR)
Computes wellhead and bottom hole pressures after rates have been determined.

CONVLV () (in file MODULE6C.FOR)
Performs numerical convolution to determine pressure drop caused by previous production.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared.**

Note: Name of the sub-program is CWATER() and the parameters passed to this sub-program are as follows:

Input Parameters:

•	*Press*	Pressure (psia)
•	*Tem*	Temperature (degree F)
•	*Sal*	Water salinity (ppm by weight)

Output Parameter:

•	*CWater*	Water compressibility (1/psi)

```
      FUNCTION CWater (Press, Tem, Sal)
```

Step 2: **Water salinity is converted from ppm to gram/liter.**

Note: A correlation presented by Rowe and Chou (Jour. Ch. and Eng. Data, V. 15, No. 1, pp. 61-66) is adapted for the water salinity conversion. The procedure uses 3 iterations to find water specific volume.

```
      T = (Tem - 32.0) / 1.8 + 273.
      AT = 5.916365 - 0.01035794 * T + 0.9270048e-5 * T * T
   +     - 1127.522 / T + 100674.1 / (T * T)
      DT = -2.5166 + 0.0111766 * T - 0.170552e-4 * T * T
      ET = 2.84851 - 0.0154305 * T + 0.223982e-4 * T * T
      X = Sal / 1.e6
      Do Iter = 1, 3
              v = AT + DT * X + ET * X * X
              X = Sal / 1.e6 / v
      End Do
```

Step 3: **Water compressibility is calculated using Osif's correlation.**

```
      Conc = X * 1000.
      F = 7.033 * Press + 54.15 * Conc - 537 * Tem + 403300.
      Cwater = 1. / F
```

Step 4: **The program control is returned back to the calling routine (sub-program WETQ(), CALCOF(), WITER(), CALCPQ(), or CONVLV()) and the sub-program CWATER() is ended.**

```
      Return
      End
```

SUB-PROGRAM PRESUR()

LOCATION: MODULE6B.FOR

MAIN THEME: This routine performs inverse table look-up of pressure given real gas potential (pseudo-pressure).

CALLS: None

CALLED BY: RATE1() (in file MODULE6B.FOR)
Calculates gas flow rates for primary well under pressure constraint.

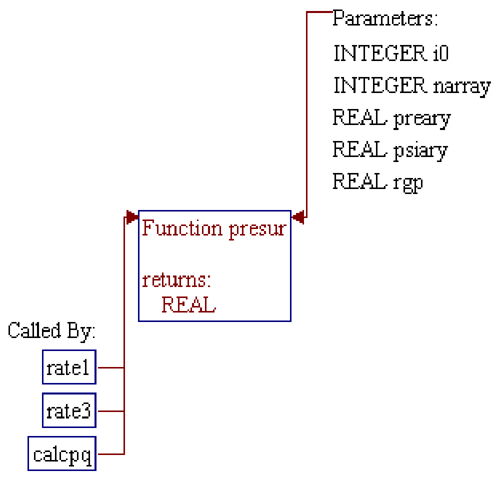
RATE3() (in file MODULE6B.FOR)
Calculates gas flow rates for infill wells (infill once) under pressure constraint.

CALCPQ() (in file MODULE6D.FOR)
Computes wellhead and bottom hole pressures after rates have been determined.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variables are declared.**

Note: Name of the sub-program is PRESUR() and the parameters passed to this sub-program are as follows:

Input Parameters:

•	*RGP*	Real gas potential or pseudo-pressure (psia**2/cp)
•	*NArray*	Number of data points in table
•	*PreAry*	Array of pressure data (psia)
•	*PsiAry*	Array of real gas potential data (psia**2/cp)

Output Parameter:

•	*Presur*	Pressure (psia)

```
         FUNCTION   Presur (RGP,    NArray, PreAry, PsiAry)
```

Note: Local variables for pressure and pseudo-pressure tables are declared.

```
         Dimension PreAry(99),PsiAry(99)
```

Step 2: **The pressure is set to zero for pseudo-pressure of zero or negative.**

```
         If (RGP .le. 0.) then
                Presur = 0.
```

Step 3: **The pressure is set to the corresponding end point values if pseudo-pressure is out of range.**

Note: If pseudo-pressure is less than the value of the first data in the table, the pressure is calculated based on the pseudo-pressure of the first data point.

```
         Else If (RGP .ge. PsiAry(NArray)) then
                Presur = PreAry(NArray)
         Else If (RGP .le. PsiAry(1)) then
                Presur = Sqrt( RGP / PsiAry(1)) * PreAry(1)
```

Step 4: **Location of data points for linear interpolation is searched using Bi-section method.**

Note: *I0* and *I1* are pointers for linear interpolation. First, these pointers are pointed to the first and last points in the table. *D* is half number of data points which is a maximum number of iteration (*IMax*) in Bi-section. This method searches for the pointer *I0* and *I1* by trial and error procedure. In each iteration, location of *I0* or *I1* is updated using the middle point of the two pointers (*I2*) which reduces the number of data to be searched by half. The iteration is continued until the pseudo-pressure (RGP) is between the values of pseudo-pressures designated by pointers *I0* and *I1*.

```
      Else
              I0 = 1
              I1 = NArray
              DN = Float(NArray)
              D  = Log(DN)/Log(2.) + 1.
              IMax = Int(D)
              Do I = 1, IMax
                     If (I1 .gt. I0 + 1) then
                            I2 = (I0 + I1 ) / 2
                            If (RGP .le. PsiAry(I2)) then
                                   I1 = I2
                            Else
                                   I0 = I2
                            End If
                     End If
              End Do
```

Step 5: **The pressure is calculated using linear interpolation.**

```
              Presur = Sqrt((RGP-PsiAry(I0))/(PsiAry(I1)-PsiAry(I0))*
     +              (PreAry(I1)**2-PreAry(I0)**2)+PreAry(I0)**2)

       End If
```

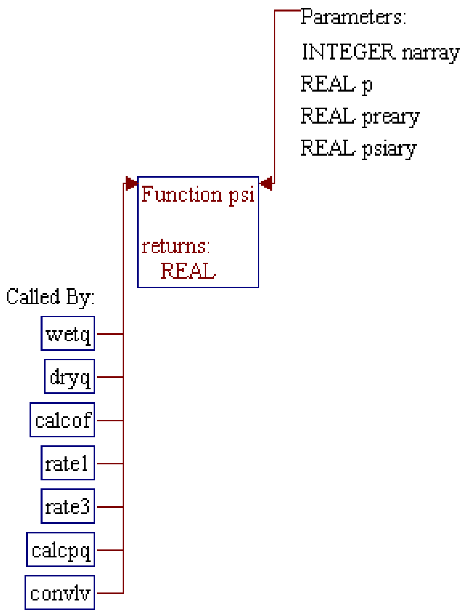
Step 6: **Program control is returned back to the calling routines (sub-program RATE1(), RATE3(), or CALCPQ()) and the sub-program PRESUR() is ended.**

```
       Return
       End
```

SUB-PROGRAM PSI()

LOCATION:	MODULE6B.FOR
MAIN THEME:	This routine performs table look-up of real gas potential (pseudo-pressure) given pressure.
CALLS:	None
CALLED BY:	WETQ() (in file MODULE6A.FOR) Calculates gas flow rates for wet coal or shale reservoirs based on bottom hole pressure. DRYQ() (in file MODULE6A.FOR) Calculates gas flow rates for dry coal and dry shale reservoirs based on bottom hole pressure. CALCOF() (in file MODULE6B.FOR) Calculates open flow potentials. RATE1() (in file MODULE6B.FOR) Calculates gas flow rates for primary well under pressure constraint. RATE3() (in file MODULE6B.FOR) Calculates gas flow rates for infill wells (infill once) under pressure constraint. CALCPQ() (in file MODULE6D.FOR) Computes wellhead and bottom hole pressures after rates have been determined. CONVLV () (in file MODULE6C.FOR) Performs numerical convolution to determine pressure drop caused by previous production.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variables are declared.**

Note: Name of the sub-program is PSI() and the parameters passed to this sub-program are as follows:

Input Parameters:

•	*P*	Pressure (psia)
•	*NArray*	Number of data points in table
•	*PreAry*	Array of pressure data (psia)
•	*PsiAry*	Array of real gas potential data (psia**2/cp)

Output Parameter:

•	*Psi*	Real gas potential or pseudo-pressure (psia**2/cp)

```
      FUNCTION   Psi    (P,NArray, PreAry, PsiAry)
```

Note: Local variables for pressure and pseudo-pressure tables are declared.

```
      Dimension PreAry(99),PsiAry(99)
```

Step 2: **The pseudo-pressure is set to zero for pressure of zero or negative.**

```
      If (P .le. 0.) then
              Psi = 0.
```

Step 3: **The pseudo-pressure is set to the end point values if pressure is higher than maximum pressure in the table.**

```
      Else If (P .ge. PreAry(NArray)) then
              Psi = PsiAry(NArray)
```

Step 4: **Location of data points for linear interpolation is directly calculated.**

Note: Since independent variable (pressure) in the table is equally spaced, direct calculation method can be used to locate the pointer for linear interpolation (*Ix*).

```
         Else
                 x = P / PreAry(1)
                 Ix = Int (x)
                 If (Ix .gt. NArray - 1) Ix = NArray – 1
```

Step 5: **The pseudo-pressure is calculated.**

Note: If pressure is located within the first two data points, the pseudo-pressure is calculated directly by multiplying the first data point value (*PsiAry(1)*) with the square of pressure to pressure-spacing ratio (*x*). Otherwise, linear interpolation is utilized.

```
                 If (x .ge. 1.) then
                         Psi = PsiAry(Ix) + (PsiAry(Ix+1) - PsiAry(Ix)) *
     +                       ( P**2 - PreAry(Ix)**2) / (PreAry(Ix+1)**2 -
     +                         PreAry(Ix)**2)
                 Else
                         Psi = PsiAry(1) * x**2
                 End If
         End If
```

Step 6: **Program control is returned back to the calling routines (sub-program WETQ(), DRYQ(), CALCOF(), RATE1(), RATE3(), CALCPQ(), or CONVLV()) and the sub-program PSI() is ended.**

```
         Return
         End
```

SUB-PROGRAM REALGS()

LOCATION: MODULE6C.FOR

MAIN THEME: This routine calculates real gas potential (pseudo-pressure), gas viscosity, and gas compressibility factor (Z-factor) as functions of pressure. The pseudo-pressure as a function pressure is calculated using the following integral equation:

$$y = 2\int \frac{p}{mZ} dp$$

where:

y	pseudo-pressure (psia**2/cp)
p	pressure (psia)
m	gas viscosity (cp)
Z	gas Z-factor

CALLS: CRIT() (in file MODULE6D.FOR)
Computes pseudo-critical properties of natural gas using Standing’s correlation with corrections for nitrogen, hydrogen sulfide, and carbon dioxide which are determined by the method of Wchert and Aziz.

VISGA() (in file MODULE6C.FOR)
Determines natural gas viscosity at a pressure of 1 atm, corrected for nitrogen, hydrogen sulfide, and carbon dioxide.

VISGR() (in file MODULE6C.FOR)
Determines reduced viscosity of natural gas.

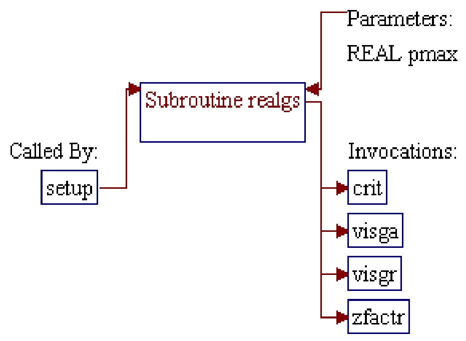
ZFACTR() (in file MODULE6D.FOR)
Calculates gas compressibility factor (Z-factor) as a function of pressure and temperature using Hall and Yarborough correlation.

CALLED BY: SETUP() (in file MODULE6C.FOR)
Sets up real gas potential (pseudo-pressure), viscosity, and Z-factor arrays for table lookup.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included.**

Note: Name of the sub-program is REALGS() and the parameter passed to this sub-program is as follows:

- *Pmax* Maximum pressure (psia)

```
      SUBROUTINE RealGs (Pmax)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'type1.h'
      include 'type2.h'
```

Step 2: **Sub-program CRIT() is invoked to calculate gas critical properties. Sub-program VISGA() is invoked to calculate gas viscosity at 1 atm.**

```
      Call CRIT(GasGrv1, CncH2S, CncCO2, CncN2, Tpc, Ppc)
      Tpr = (Tem + 460.) / Tpc
      Va  = VisGa (GasGrv1, Tem, CncH2S, CncCO2, CncN2)
```

Step 3: **Some variables and arrays are initialized.**

```
      Do I = 1, NArray
              PreAry(I) = Float(I) * Pmax / Float(NArray)
              PsiAry(I) = 0.
      End Do
      P1   = 0.0
      V1   = 1.0
      Z1   = 1.0
      Ppsi = 0.0
      F1   = 0.0
      DP   = PreAry(1)
```

Step 4: **Pseudo-pressure is calculated using Simpson's rule (numerical integration) with one intermediate pressure point between each tabulated point.**

Note: Gas reduced viscosity and Z-factor are calculated in sub-programs VISGR() and ZFACTR(), respectively.

```
         Do I = 1, NArray
                P2  = (PreAry(I) + P1) / 2.
                Ppr = P2 / Ppc
                V2  = VisGr (Tpr, Ppr)
                Z2  = ZFactr(Tpr, Ppr)
                F2  = 2. * P2 / (V2 * Va * Z2)
                P3  = PreAry(I)
                Ppr = P3 / Ppc
                V3  = VisGr (Tpr, Ppr)
                Z3  = ZFactr(Tpr, Ppr)
                F3  = 2. * P3 / (V3 * Va * Z3)
                Ppsi = Ppsi + (F1 + 4. * F2 + F3) / 6. * DP
                PsiAry(I) = Ppsi
                VisAry(I) = V3 * Va
                ZAry(I)   = Z3
                P1 = P3
                F1 = F3
         End Do
```

Step 5: **The program control is returned back to the calling routine (sub-program SETUP()) and the sub-program REALGS() is ended.**

```
         Return
         End
```

SUB-PROGRAM RELPRM()

LOCATION: MODULE6A.FOR

MAIN THEME: This routine calculates gas and water relative permeabilities for wet and shale reservoirs. All relative permeability relationships are based on simulation results for multiple wells in specific geographical regions:

• Eastern Coal	From rock creek project, Alabama
• Western Coal	From northern San Juan Basin, Colorado
• Shale	From Otsego County, Michigan

Table look-up method with linear interpolation is implemented for relative permeability calculations.

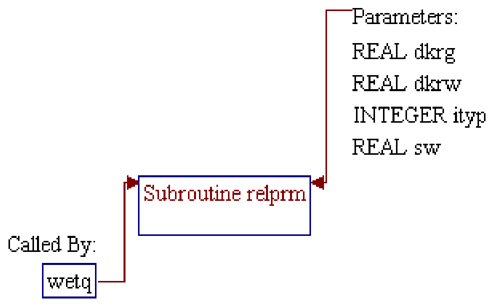
CALLS: None

CALLED BY: WETQ() (in file MODULE6A.FOR)
Calculates gas flow rates for wet coal or shale reservoirs based on bottom hole pressure.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variables are declared.**

Note: Name of the sub-program is RELPRM() and the parameters passed to this sub-program are as follows:

• *Sw*	Water saturation, fraction
• *ITyp*	Code for location/type of reservoir: 1=Eastern Coal, 2=Western Coal, 3=Antrim Shale
• *DKrg*	Gas relative permeability, fraction
• *DKrw*	Water relative permeability, fraction

```
      SUBROUTINE RelPrm (Sw,ITyp,DKrg,DKrw)
```

Note: Local variables for tables of relative permeability data are declared.

```
      Dimension GasTbl(7,3), WtrTbl(7,3), SwTbl(7,3)
```

Step 2: **Data for relative permeability tables are assigned to variables *GasTbl()* (gas relative permeability), *WtrTbl()* (water relative permeability), and *SwTbl()* (water saturation).**

```
      Data ((GasTbl(I,J),I=1,7),J=1,3)/
     + 0.700, 0.520, 0.300, 0.175, 0.080, 0.030, 0.000,
     + 0.350, 0.200, 0.120, 0.080, 0.060, 0.050, 0.000,
     + 0.700, 0.420, 0.270, 0.150, 0.060, 0.020, 0.000/
      Data ((WtrTbl(I,J),I=1,7),J=1,3)/
     + 0.000, 0.005, 0.045, 0.120, 0.300, 0.630, 1.000,
     + 0.000, 0.050, 0.100, 0.200, 0.400, 0.700, 1.000,
     + 0.000, 0.010, 0.020, 0.080, 0.170, 0.380, 1.000/
      Data ((SwTbl(I,J),I=1,7),J=1,3)/
     + 0.450, 0.500, 0.600, 0.700, 0.800, 0.900, 1.000,
     + 0.000, 0.500, 0.750, 0.850, 0.900, 0.950, 1.000,
     + 0.400, 0.500, 0.600, 0.700, 0.800, 0.900, 1.000/
```

Step 3: **Table look-up procedure with linear interpolation is performed to calculate gas and water relative permeabilities (*DKrg* and *DKrw*) where their values are forced between the span of relative permeability data.**

```
      If (Sw .le. SwTbl(1,ITyp)) then
          DKrw = WtrTbl(1,ITyp)
          DKrg = GasTbl(1,ITyp)
      Else If (Sw .ge. SwTbl(7,ITyp)) then
          DKrw = WtrTbl(7,ITyp)
          DKrg = GasTbl(7,ITyp)
```

```
         Else
            Do J = 2, 7
               If ((Sw.ge.SwTbl(J-1,ITyp)) .and.
     +             (Sw.lt.SwTbl(J,ITyp))) then
                  X    = (Sw-SwTbl(J-1,ITyp))/
     +                   (SwTbl(J,ITyp)-SwTbl(J-1,ITyp))
                  DKrw =  WtrTbl(J-1,ITyp) + X *
     +                   (WtrTbl(J,ITyp)-WtrTbl(J-1,ITyp))
                  DKrg =  GasTbl(J-1,ITyp) + X *
     +                   (GasTbl(J,ITyp)-GasTbl(J-1,ITyp))
               End If
            End Do
         End If
```

Step 4: **The program control is returned back to the calling routine (sub-program WETQ()) and the sub-program RELPRM() is ended.**

```
         Return
         End
```

SUB-PROGRAM RHOW()

LOCATION: MODULE6C.FOR

MAIN THEME: This routine calculates water density using a curve fit to a correlation presented in the Petroleum Engineers Handbook.

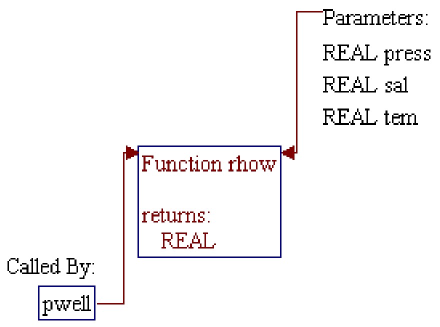
CALLS: None

CALLED BY: PWELL() (in file MODULE6B.FOR)
Calculates bottomhole pressure, wellhead pressure, or flow rate based on the difference between well head pressure and bottom hole pressure of the well using Smith's formula.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared.**

Note: Name of the sub-program is RHOW() and the parameters passed to this sub-program are as follows:

• *Press*	Pressure (psia)
• *Tem*	Temperature (degree F)
• *Sal*	Water salinity (ppm by weight)
• *RhoW*	Water density (gr/cc)

```
      FUNCTION   RhoW   (Press,  Tem,    Sal)
```

Step 2: **Water density as a function of pressure, temperature, and water salinity is calculated.**

```
      X = Sal / 120000.
      Rho1 = (1.001125 + 0.095062*X + 0.001688*X*X) +
     +       (1.25     - 20.0    *X + 3.75    *X*X) *Tem    *1.e-5+
     +       (-10.15625+ 5.859375*X -1.171875 *X*X) *Tem**2 *1.e-7
      DRho = 0.0226 * Press / 6000. / Rho1 ** 1.3
      RhoW = Rho1 + DRho
```

Step 3: **The program control is returned back to the calling routine (sub-program PWELL()) and the sub-program RHOW() is ended.**

```
      Return
      End
```

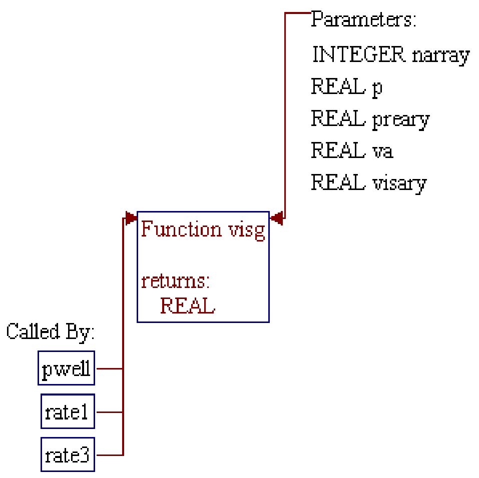
SUB-PROGRAM VISG()

LOCATION:	MODULE6C.FOR
MAIN THEME:	This routine performs table look-up of gas viscosity as a function of pressure using linear interpolation.
CALLS:	None

CALLED BY:	PWELL() (in file MODULE6B.FOR)
	Calculates bottomhole pressure, wellhead pressure, or flow rate based on the difference between well head pressure and bottom hole pressure of the well using Smith's formula.
	
	RATE1() (in file MODULE6B.FOR)
	Calculates gas flow rates for primary wells under pressure constraint.
	
	RATE3() (in file MODULE6B.FOR)
	Calculates gas flow rates for infill wells (infill once) under pressure constraint.

READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Local variables are declared.**

Note: Name of the sub-program is VISG() and the parameters passed to this sub-program are as follows:

• *P*	Pressure (psia)
• *NArray*	Number of points in table
• *PreAry*	Array of pressure data (psia)
• *VisAry*	Array of gas viscosity data (cp)
• *Va*	Gas viscosity at 1 atm (cp)
• *VisG*	Calculated gas viscosity (cp)

```
      FUNCTION   Visg (P, NArray, PreAry, VisAry, Va)
```

Note: Local variables are declared.

```
      Dimension PreAry(99),VisAry(99)
```

Step 2: **Gas viscosity is set to end point values if pressure is out of range.**

```
      If (P .le. 0.) then
              Visg = Va
      Else If (P .ge. PreAry(NArray)) then
              Visg = VisAry(NArray)
```

Step 3: **Location of data points for linear interpolation is directly calculated.**

Note: Since independent variable (pressure) in the table is equally spaced, direct calculation method can be used to locate the pointer for linear interpolation (*Ix*).

```
      Else
              x = P / PreAry(1)
              Ix = Int (x)
              If (Ix .gt. NArray - 1) Ix = NArray - 1
```

Step 4: **Gas viscosity is calculated.**

Note: If pressure is located within the first two data points, the gas viscosity is calculated directly based on pressure to pressure-spacing ratio (*x*). Otherwise, linear interpolation is utilized.

```
            If (x .ge. 1.) then
                    Visg = VisAry(Ix) + (VisAry(Ix+1)-VisAry(Ix)) *
     +                   ( x - PreAry(Ix)/PreAry(1))
            Else
                    Visg = Va + (VisAry(1) - Va) * x
            End If
    End If
```

Step 5: **Program control is returned back to the calling routine (sub-program PWELL(), RATE1(), or RATE3()) and the sub-program VISG() is ended.**

```
    Return
    End
```

SUB-PROGRAM VISGA()

LOCATION: MODULE6C.FOR

MAIN THEME: This routine determines natural gas viscosity at a pressure of 1 atm, corrected for nitrogen, hydrogen sulfide, and carbon dioxide. The function used for calculation was adapted from a program presented in the ERCB manual. The acceptable ranges of the curve fits are: gas gravity from 0.55 to 1.5, concentration of CO2, H2S, or N2 between 0 and 0.15, and temperature between 40 and 400 degrees F.

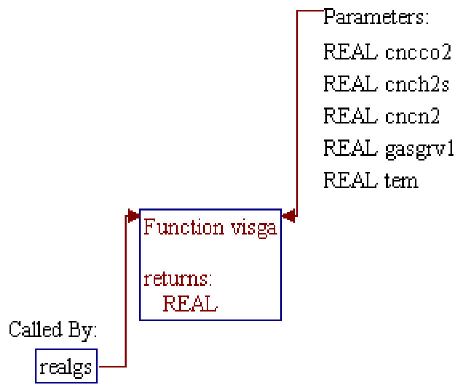
CALLS: None

CALLED BY: REALGS() (in file MODULE6C.FOR)
Calculates real gas potential (pseudo-pressure), gas viscosity, and gas compressibility factor (Z-factor) as functions of pressure.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared.**

Note: Name of the sub-program is VISGA() and the parameters passed to this sub-program are as follows:

• *GasGrv1*	Gas specific gravity (Air=1)
• *Tem*	Gas temperature (degree F)
• *CncH2S*	Concentration of hydrogen sulfide (fraction)
• *CncCO2*	Concentration of carbon dioxide (fraction)
• *CncN2*	Concentration of nitrogen (fraction)
• *VisGa*	Calculated gas viscosity at 1 atm (cp)

```
      FUNCTION   VisGa  (GasGrv1, Tem, CncH2S, CncCO2, CncN2)
```

Step 2: **Gas gravity and temperature are adjusted to allowable limit.**

```
      G = Min(GasGrv1,1.5)
      G = Max(G,0.5)
      T = Min(Tem,400.)
      T = Max(T,40.)
```

Step 3: **Uncorrected gas viscosity at 1 atm is calculated.**

```
      VisGu  = 0.0126585 - 0.611823e-02 * G + 0.164574e-02 *
     +           G * G + 0.164574e-04 * T - 0.719221e-06 *
     +           G * T - 0.609046e-06 * G * G * T
```

Step 4: **Gas viscosity corrections due to H2S, CO2, or N2 content(s) are calculated.**

Note: Prior to the calculations, concentration of the impurities are adjusted to allowable limits.

```
      C = Min(CncH2S,0.15)
      C = Max(C,0.)
      CorH2S = (0.000113 * C*100. * G - 0.000038 * C*100. +
     +           0.000001) * (1.0/(1.0 + G)) + 0.000001
      C = Min(CncCO2,0.15)
      C = Max(C,0.)
      CorCO2 = (0.000134 * C*100. * G - 0.000004 * C*100. +
     +           0.000004 * G) * (1.0/(1.0 + G)) - 0.000003
      C = Min(CncN2,0.15)
      C = Max(C,0.)
      CorN2  = (0.000170 * C*100. * G - 0.000021 * C*100. +
     +           0.000010 * G)* (1.0/(1.0 + G)) - 0.000006
```

Step 5: **Gas viscosity at 1 atm is modified to account for impurities.**

```
          VisGa  = VisGu + CorH2S + CorCO2 + CorN2
```

Step 6: **Program control is returned back to the calling routine (sub-program REALGS()) and the sub-program VISGA() is ended.**

```
          Return
          End
```

SUB-PROGRAM VISGR()

LOCATION: MODULE6C.FOR

MAIN THEME: This routine determines reduced viscosity of natural gas. The function used in the calculation is adapted from a program presented in the ERCB manual. The reduced viscosity is the gas viscosity at a given temperature divided by the gas viscosity at one atmosphere and at the given temperature. The ERCB data were adjusted to better match the Carr, et al. correlation.

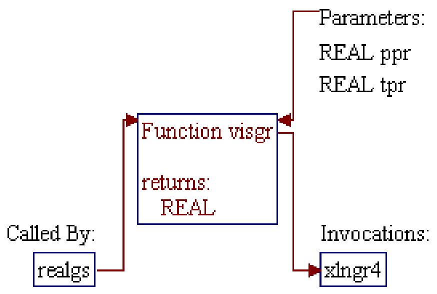
CALLS: XLNGR4() (in file MODULE6C.FOR)
Performs four-point lagrange interpolation for sub-program VISGR() to interpolate viscosity ratio based on pseudo-reduced temperature.

CALLED BY: REALGS() (in file MODULE6C.FOR)
Calculates real gas potential (pseudo-pressure), gas viscosity, and gas compressibility factor (Z-factor) as functions of pressure.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Local variables are declared.**

Note: Name of the sub-program is VISGR() and the parameters passed to this sub-program are as follows:

• *Tpr*	Pseudo-reduced temperature
• *Ppr*	Pseudo-reduced pressure
• *VisGr*	calculated reduced gas viscosity

```
      FUNCTION   VisGr(Tpr, Ppr)
```

Note: Local variables are declared.

```
      Dimension TemTbl(13), PrsTbl(22), VisTbl(22,13)
```

Step 2: **Data of pseudo reduced temperature, pressure, and gas viscosity are assigned to array variables *TemTbl()*, *PrsTbl()*, and *VisTbl()*, respectively.**

Note: These data will be utilized in the Langrange interpolation equation in sub-program XLNGR4().

```
      Data TemTbl /1.05,1.10,1.15,1.20,1.30,1.40,1.50,1.60,1.75,2.00,
     +        2.25,2.50,3.00/
      Data PrsTbl /0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0,1.2,1.4,
     +        1.6,1.8,2.0,3.0,4.0,6.0,8.0,10.0,15.0,20.0/
      Data ((VisTbl(I,J),I=1,22),J=1,7)/
     +1.000,1.012,1.025,1.050,1.075,1.100,1.145,1.195,1.285,1.400,1.760,
     +2.285,2.865,3.290,3.800,4.760,5.500,6.500,7.250,7.900,9.080,9.850,
     +1.000,1.011,1.023,1.043,1.065,1.086,1.120,1.150,1.195,1.255,1.435,
     +1.700,2.070,2.465,2.800,3.850,4.655,5.720,6.500,7.100,8.260,9.000,
     +1.000,1.010,1.021,1.036,1.055,1.073,1.095,1.120,1.145,1.175,1.280,
     +1.420,1.590,1.850,2.100,3.225,3.975,5.030,5.820,6.385,7.550,8.250,
     +1.000,1.009,1.019,1.030,1.045,1.060,1.070,1.085,1.110,1.135,1.195,
     +1.285,1.425,1.570,1.750,2.600,3.350,4.380,5.200,5.740,6.900,7.600,
     +1.000,1.008,1.017,1.027,1.040,1.054,1.063,1.075,1.100,1.115,1.155,
     +1.215,1.285,1.360,1.450,2.020,2.560,3.490,4.185,4.700,5.790,6.500,
     +1.000,1.007,1.015,1.024,1.035,1.048,1.056,1.067,1.089,1.100,1.135,
     +1.185,1.235,1.280,1.335,1.680,2.100,2.790,3.380,3.860,4.790,5.410,
     +1.000,1.006,1.013,1.021,1.030,1.042,1.049,1.059,1.078,1.087,1.120,
     +1.150,1.185,1.220,1.260,1.500,1.785,2.325,2.820,3.230,4.060,4.610/
      Data ((VisTbl(I,J),I=1,22),J=8,13)/
     +1.000,1.005,1.011,1.018,1.025,1.036,1.042,1.051,1.067,1.075,1.108,
     +1.134,1.160,1.180,1.215,1.385,1.595,2.020,2.425,2.790,3.490,4.025,
     +1.000,1.004,1.009,1.015,1.021,1.030,1.035,1.043,1.056,1.065,1.090,
     +1.110,1.125,1.145,1.165,1.295,1.435,1.780,2.070,2.375,2.990,3.490,
     +1.000,1.003,1.007,1.012,1.017,1.024,1.028,1.035,1.045,1.050,1.060,
     +1.070,1.080,1.095,1.110,1.200,1.290,1.500,1.710,1.950,2.460,2.875,
     +1.000,1.002,1.005,1.009,1.013,1.018,1.021,1.027,1.034,1.037,1.045,
     +1.055,1.065,1.075,1.085,1.145,1.210,1.340,1.485,1.665,2.085,2.460,
     +1.000,1.001,1.003,1.006,1.009,1.012,1.015,1.019,1.023,1.025,1.030,
     +1.040,1.050,1.060,1.065,1.110,1.155,1.245,1.355,1.485,1.830,2.150,
     +1.000,1.000,1.001,1.003,1.005,1.007,1.009,1.011,1.013,1.015,1.020,
     +1.025,1.030,1.035,1.040,1.060,1.095,1.140,1.190,1.265,1.495,1.730/
```

Step 3: **Set *VisGr* equals to 1 if input parameters are out of range.**

```
        If (Tpr.lt.1.02 .OR. Tpr.gt.3.01 .OR. Ppr.lt.0.01) then
                VisGr  = 1.00
                Return
        End If
```

Step 4: **Location of data points for interpolations are located.**

```
        J = 12
        DO J1 = 11, 3, -1
                If (TemTbl(J1) .ge. Tpr) J=J1
        End Do
        I = 22
        Do I1 = 21, 2, -1
                If (PrsTbl(I1) .ge. Ppr) I=I1
        End Do
```

Step 5: **Sub-program XLNGR4() is utilized to interpolate on temperature. Linear interpolation on (*1/Ppr+1*) is used to interpolate on pressure.**

```
        Call XLNGR4 (Tpr, TemTbl(J-2), TemTbl(J-1), TemTbl(J),
     +          TemTbl(J+1), VisJ, VisTbl(I,J-2), VisTbl(I,J-1),
     +          VisTbl(I,J), VisTbl(I,J+1))
        Call XLNGR4 (Tpr, TemTbl(J-2), TemTbl(J-1), TemTbl(J),
     +          TemTbl(J+1), VisI, VisTbl(I-1,J-2), VisTbl(I-1,J-1),
     +          VisTbl(I-1,J), VisTbl(I-1,J+1))
        VisGr = VisI + ((1./(1.+Ppr) - 1./(1.+PrsTbl(I-1)))/
     +          (1./(1.+PrsTbl(I)) - 1./(1.+PrsTbl(I-1))))*(VisJ-VisI)
```

Step 6: **Program control is returned back to the calling routine (sub-program REALGS()) and the sub-program VISGR() is ended.**

```
        Return
        End
```

SUB-PROGRAM VISW()

LOCATION: MODULE6C.FOR

MAIN THEME: This routine calculates water viscosity using Meehan's correlation.

CALLS: None

CALLED BY: WETQ() (in file MODULE6A.FOR)
Calculates gas flow rates for wet coal or shale reservoirs based on bottom hole pressure.

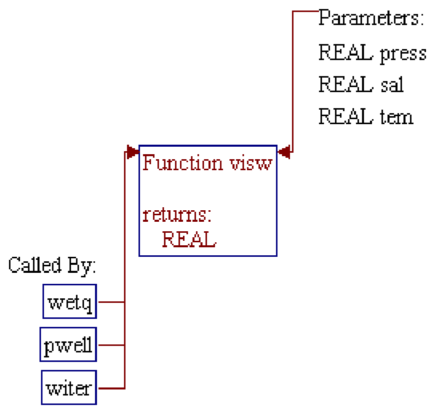
PWELL() (in file MODULE6B.FOR)
Calculates bottomhole pressure, wellhead pressure, or flow rate based on the difference between well head pressure and bottom hole pressure of the well using Smith's formula.

WITER() (in file MODULE6C.FOR)
Computes gas production as a function of average reservoir pressure, based on material balance for water drive reservoirs.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared.**

Note: Name of the sub-program is VISW() and the parameters passed to this sub-program are as follows:

• *Press*	Pressure (psia)
• *Tem*	Temperature (degree F)
• *Sal*	Water salinity (ppm by weight)
• *VisW*	Water viscosity (cp)

```
      FUNCTION   VisW (Press, Tem, Sal)
```

Step 2: **Water viscosity is calculated using Meehan's correlation.**

Note: *Conc* is water salinity converted from ppm to fraction.

```
      Conc = Sal * 0.0001
      SqConc = Sqrt(Conc)
      Sc2 = 1. + 0.00187 * SqConc + 0.000218 * SqConc * Conc * Conc +
     +     ( Sqrt(Tem) - 0.0135 * Tem) * (0.00276 - 0.000344 * SqConc)*
     +       Conc
      Sp = 1. + 3.5e-12 * Press * Press * (Tem - 40.)
      VisW  = Sc2 * Sp * 0.02414 * 10. ** (446.04 / (Tem + 208.))
```

Step 3: **Program control is returned back to the calling routine (sub-program WETQ(), PWELL(), or WITER()) and the sub-program VISW() is ended.**

```
      Return
      End
```

SUB-PROGRAM XLNGR4()

LOCATION: MODULE6C.FOR

MAIN THEME: This routine performs four-point lagrange interpolation for sub-program VISGR() to interpolate viscosity ratio based on pseudo-reduced temperature. The procedure was adapted from ERCB manual. The general Lagrange equation (K.L. Neilson Methods in Numerical Analysis, the MacMillan Company, 1956) is solved for the dependent parameter (Y) value corresponding to a given independent parameter (X) lying in the range of four data points: (X1,Y1), (X2,Y2), (X3,Y3), and (X4,Y4).

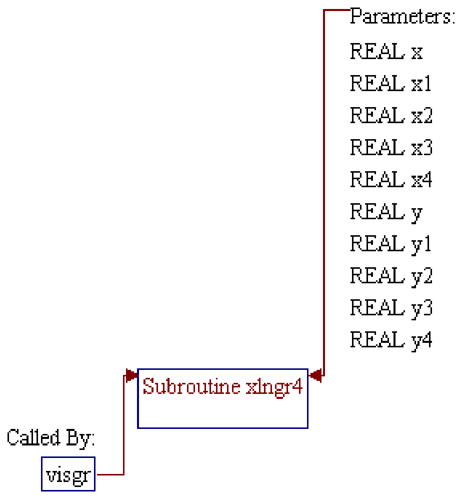
CALLS: None

CALLED BY: VISGR() (in file MODULE6C.FOR)
Determines reduced viscosity of natural gas.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared.**

Note: Name of the sub-program is XLNGR4() and the parameters passed to this sub-program are as follows:

Input Parameters:

• *X*	X-coordinate of desired point
• *X1,X2,X3,X4*	X-coordinate of tabulated data
• *Y1,Y2,Y3,Y4*	Y-coordinate of tabulated data

Output Parameter:

• *Y*	Computed Y-coordinate for specified X

```
      SUBROUTINE XLngr4 (x, x1, x2, x3, x4, y, y1, y2, y3, y4)
```

Step 2: **Dependent variable *Y* as a function of *X* is calculated using four point Lagrange interpolation.**

```
            a1 = x1 - x2
            a2 = x1 - x3
            a3 = x1 - x4
            a4 = x2 - x3
            a5 = x2 - x4
            a6 = x3 - x4
            b1 = x  - x1
            b2 = x  - x2
            b3 = x  - x3
            b4 = x  - x4
       y  = b2 / a1 * b3 / a2 * b4 / a3 * y1 -
     +      b1 / a1 * b3 / a4 * b4 / a5 * y2 +
     +      b1 / a2 * b2 / a4 * b4 / a6 * y3 -
     +      b1 / a3 * b2 / a5 * b3 / a6 * y4
```

Step 3: **The program control is returned back to the calling routine (sub-program VISGR()) and the sub-program XLNGR4() is ended.**

```
      Return
      End
```

SUB-PROGRAM ZEE()

LOCATION: MODULE6C.FOR

MAIN THEME: This routine performs table look-up linear interpolation of Z-factor as a function of pressure.

CALLS: None

CALLED BY: WETQ() (in file MODULE6A.FOR)
Calculates gas flow rates for wet coal or shale reservoirs based on bottom hole pressure.

DRYQ() (in file MODULE6A.FOR)
Calculates gas flow rates for dry coal and dry shale reservoirs based on bottom hole pressure.

CALCOF() (in file MODULE6B.FOR)
Calculates open flow potentials.

PWELL() (in file MODULE6B.FOR)
Calculates bottomhole pressure, wellhead pressure, or flow rate based on the difference between well head pressure and bottom hole pressure of the well using Smith's formula.

RATE1() (in file MODULE6B.FOR)
Calculates gas flow rates for primary wells under pressure constraint.

RATE3() (in file MODULE6B.FOR)
Calculates gas flow rates for infill wells (infill once) under pressure constraint.

SETUP() (in file MODULE6C.FOR)
Sets up real gas potential (pseudo-pressure), viscosity, and Z-factor arrays for table lookup and calculates original gas in place.

WITER() (in file MODULE6C.FOR)
Calculates water influx and based on that decides whether production from a specific water drive reservoir needs to be stopped (if water fills up the reservoir) or not.

CALCPQ() (in file MODULE6D.FOR)
Computes wellhead and bottom hole pressures after rates have been determined.

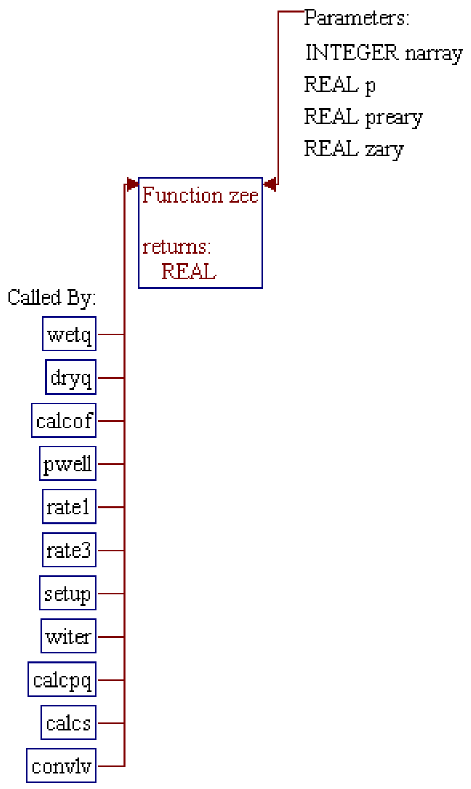
CALCS() (in file MODULE6D.FOR)
Performs numerical convolution and solves for pressure and flow rates at each time step to generate type curves.

CONVLV () (in file MODULE6C.FOR)
Performs numerical convolution to determine pressure drop caused by previous production.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Local variables are declared.**

Note: Name of the sub-program is ZEE() and the parameters passed to this sub-program are as follows:

Input Parameters:

• *P*	Pressure (psia)
• *NArray*	Number of data points in table
• *PreAry*	Array of pressure data (psia)
• *ZAry*	Array of Z-factor data

Output Parameter:

• *Zee*	Computed Z-factor

```
FUNCTION Zee (P, NArray, PreAry, ZAry)
```

Note: Local variables for pressure and Z-factor tables are declared.

```
Dimension PreAry(99),ZAry(99)
```

Step 2: **Gas Z-factor is set to end point values if pressure is out of range.**

```
If (P .le. 0.) then
        Zee = 1.
Else If (P .ge. PreAry(NArray)) then
        Zee = ZAry(NArray)
```

Step 3: **Location of data points for linear interpolation is directly calculated.**

Note: Since independent variable (pressure) in the table is equally spaced, direct calculation method can be used to locate the pointer for linear interpolation (*Ix*).

```
Else
        x = P / PreAry(1)
        Ix = Int (x)
        If (Ix .gt. NArray - 1) Ix = NArray - 1
```

Step 4: **Gas Z-factor is calculated.**

Note: If pressure is located within the first two data points, the gas Z-factor is calculated directly based on pressure to pressure-spacing ratio (*x*). Otherwise, linear interpolation is utilized.

```
               If (x .ge. 1.) then
                       Zee = ZAry(Ix) + (ZAry(Ix+1) - ZAry(Ix)) *
     +                     ( x - PreAry(Ix)/PreAry(1))
               Else
                       Zee = 1. + (ZAry(1) - 1.) * x
               End If
       End If
```

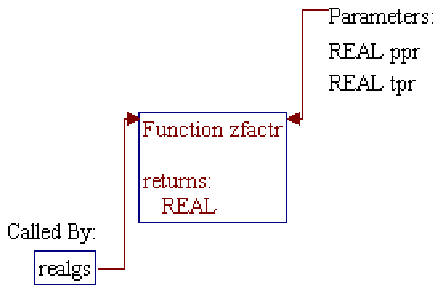
Step 5: **Program control is returned back to the calling routine (sub-program WETQ(), DRYQ(), CALCOF(), PWELL(), RATE1(), RATE3(), SETUP(), WITER(), CALPQ(), CALCS(), or CONVLV()) and the sub-program ZEE() is ended.**

```
       Return
       End
```

SUB-PROGRAM ZFACTR()

LOCATION:	MODULE6D.FOR
MAIN THEME:	This routine calculates gas compressibility factor (Z-factor) as a function of pressure and temperature using Hall and Yarborough correlation. A constrained Newton-Raphson procedure is used to solve the equation of state.
CALLS:	None
CALLED BY:	REALGS() (in file MODULE6C.FOR) Calculates real gas potential (pseudo-pressure), gas viscosity, and gas compressibility factor (Z-factor) as functions of pressure.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared.**

Note: Name of the sub-program is ZFACTR() and the parameters passed to this sub-program are as follows:

• *Tpr*	Pseudo-reduced temperature
• *Ppr*	Pseudo-reduced pressure
• *ZFactr*	Calculated gas Z-factor

```
      FUNCTION ZFactr (Tpr, Ppr)
```

Step 2: **Gas Z-factor is set to 1 if pseudo-reduced temperature and pressure are out of range.**

```
      If ((Tpr.lt.0.9999) .or. (Ppr.le.0.001)) then
              ZFactr = 1.0
              Return
      End If
```

Step 3: **Newton-Raphson iteration is used to solve for gas Z-factor from the Hall and Yarborough correlation.**

Note: A maximum of 10 iterations is performed with objective function tolerance of 0.00001. Parameter *y* in the objective function (*F*) is a reduced density.

```
      A = 0.06125 / Tpr  * Exp (-1.2 * (1.0 - 1.0 / Tpr)**2 )
      B = 14.76   / Tpr  - 9.76  / Tpr**2 + 4.58 / Tpr**3
      C = 90.7    / Tpr  - 242.2 / Tpr**2 + 42.4 / Tpr**3
      D = 2.82    / Tpr  + 2.18
      y  = 0.001
      F  = 1.
      dy = 1.
      Do IZ = 1, 10
              If (abs(F) .gt. 0.00001) then
                      F  = - A*Ppr + ( y + y**2 + y**3 - y**4 )
     +                         / (1. - y)**3 - B*y*y + C*y**D
                      Fp = ( 1. + 4.*y + 4.*y**2 - 4.*y**3 +y**4 )
     +                         / (1. - y)**4 - 2*B*y + C*D*y**(D-1)
                      dy = F/Fp
                      y  = y - dy
                      If (y .gt. 0.6) y = 0.6
                      If (y .lt. 0.1e-5) y = 0.1e-5
              End If
      End Do
      ZFactr = A * Ppr / y
```

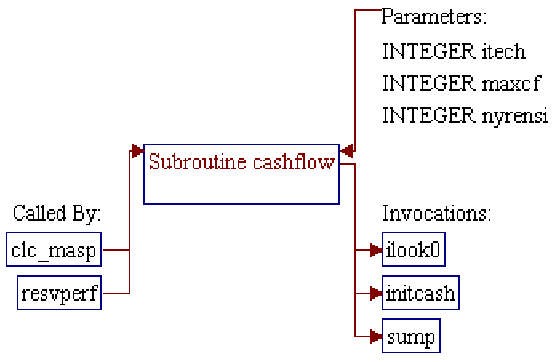
Step 4: **Program control is returned back to the calling routine (sub-program REALGS()) and the sub-program ZFACTR() is ended.**

```
            Return
            End
```

SUB-PROGRAM CASHFLOW()

LOCATION:	CASHFLOW.FOR
MAIN THEME:	This routine performs a discounted cash flow analysis for every gas reservoir (i.e. performs a pro-forma cash flow analysis for every reservoir processed)
CALLS:	ILOOK0() (in file IOFUNCT.FOR) Searches location of an integer number in a set of array.
	INITCASH (in file INITIAL.FOR) Initializes cash flow variables as declared in header file CASHFLOW.H.
	SUMP() (in file IOFUNCT.FOR) **A**dds all numbers in a set of a real array.
CALLED BY:	CLC_MASP() (in file CLC_MASP.FOR) Calculates Minimum Acceptable Supply Price (MASP) of a specified development type in a specified pay grade.
	RESVPERF (in file RESVPERF.FOR) Main program of Reservoir Performance Module.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameter of the sub-program are declared. Header ".h" files are included. Local variables are declared.**

Note: Name of the sub-program is CASHFLOW() and the parameter passed to this sub-program is as follows:

• *itech*	Technology flag (1=current, 2=advanced)
• *nyrensi*	Not currently used
• *maxcf*	Flag for environmental RP run: 0=non-environmental RP run, 1=environmental RP run (not currently used)

```
      subroutine cashflow(itech,nyrensi,maxcf)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'global.h'
      include 'cashflow.h'
      include 'costing.h'
      include 'cost.h'
      include 'field.h'
      include 'tax_nat.h'
      include 'tax_reg.h'
      include 'unitcost.h'
      include 'gsamvar.h'
```

Note: Local variables are declared.

```
      real*4 schedule(8)
      integer iyr,iyr1,itech
      real*4 temp
      real*4 tgglcd(qyr)
```

Step 2: **Depreciation schedule data is assigned. (MACRS Schedule)**

```
      data schedule/0.1423,0.2449,0.1749,0.1249,
     &    0.0893,0.0893,0.0893,0.0446/
```

Step 3: **Sub-program INITCASH is invoked to initialize cash flow variables.**

Note: The cash flow variables are declared in header file CASHFLOW.H.

```
        call initcash
```

Step 4: **Sub-program ILOOK0() is invoked to search for State ID given in variable *state* in state tax array *tax_st()* and stores the pointer to variable *istate.***

Note: If no match is found (*istate=0*), the default location/pointer (*qstate+1*) is utilized.

```
        call ilook0(state,tax_st,ntax_st,istate)
        if(istate.eq.0) istate=qstate+1
```

Step 5: **Array of total G&G lease cost depletion (*tgglcd()*) is initialized to zero.**

```
        do iabb=1,40
          tgglcd(iabb)=0.0
        enddo
```

Step 6: **Lease acquisition cost (*la()*) for undiscovered reservoirs (*gsamid(3:3)='1'*) or for discovered/undeveloped reservoirs (*gsamid(3:3)='2'*) is calculated.**

Note: *gsamid()* is a string of 11-digit GSAM ID, *lbc_frac* is lease bonus cost factor, *sump(gasprod,nyr)* is total gas production upto year *nyr*, *gasprod()* is gas production, *nyr* is number of simulation years, *gprice()* is gas price, and *royrate* is royalty rate. Lease cost is assigned in the first year of simulation.

```
            if (gsamid(3:3).eq.'1'.or.gsamid(3:3).eq.'2') Then
            la(1)=lbc_frac*sump(gasprod,nyr)*gprice(1)*(1-royrate)
            Endif
```

Step 7: **Federal income tax rate (*fedrate*) is assigned.**

Note: For Alberta, British Columbia, or Eastern Canada, *gasamid(3:3)='22'*, *'23'*, or *'24'*, respectively, Canada federal income tax rate (*fedrate_can*) is used. For other GSAM supply regions, U.S. federal income tax rate (*fedrate_us*) is used.

```
            fedrate = fedrate_us
            if (gsamid(1:2).eq.'22'.or.gsamid(1:2).eq.'23'.or.
      @       gsamid(1:2).eq.'24') fedrate = fedrate_can
```

Step 8: **Loop of years for cash flow calculation is initialized.**

```
       do iyr=1,nyr
```

Step 9: **Tangible development and exploratory well costs (*tang_dwc()* and *tang_ewc()*) and intangible development and exploratory well costs (*intang_dwc()* and *intang_ewc()*) in each year are calculated. Variables tang_m, intang_m and oam_m are tangible cost, intangible cost and O&M cost multipliers. These are calculated in program UNITCOST.FOR as a function of gas price.**

```
        tang_dwc(iyr)=dwc(iyr)*dwc_tan(itech)*tang_m(iyr)
        tang_ewc(iyr)=ewc(iyr)*ewc_tan(itech)*tang_m(iyr)
        intang_dwc(iyr)=dwc(iyr)*(1-dwc_tan(itech))*intang_m(iyr)
        intang_ewc(iyr)=ewc(iyr)*(1-ewc_tan(itech))*intang_m(iyr)
```

Step 10: **Adjusted gross sales (*adjgross()*), net sales (*netsales()*), G&A on expensed items (*ga_exp()*), and intangible investment (*ii()*) in each year are calculated.**

```
        adjgross(iyr)=oilprod(iyr)*oprice(iyr)+
      &   gasprod(iyr)*gprice(iyr)-gravpen(iyr)-transcst(iyr)
        netsales(iyr)=adjgross(iyr)-adjgross(iyr)*royrate
        ga_exp(iyr)=ga_exp_m(itech)*
      &   (inj(iyr)+oam(iyr)+eoam(iyr))
        ii(iyr)=
      &   intang_ewc(iyr)+intang_dwc(iyr)+icap(iyr)+eicap(iyr)
```

Step 11: **Intangible capitalized (*intcap()*) in each year is calculated.**

Note: First, *intcap()* for drilling cost is calculated if it is requested in input file TAX_NAT.DAT (*cidc=.true.*). The *intcap()* is then modified if environmental and/or other intangibles are also requested to be capitalized (*ce=.true.* and/or *coi=.true.*) in input file TAX_NAT.DAT.

```
        if(cidc) intcap(iyr)=
      &   intcap(iyr)+piic*intang_dwc(iyr)+piic*intang_ewc(iyr)
        if(ce) intcap(iyr)=intcap(iyr)+piic*eicap(iyr)
        if(coi) intcap(iyr)=intcap(iyr)+piic*icap(iyr)
```

Step 12: **Tangible investment (*ti()*) and total capitalized investment (*tci()*) are calculated.**

```
        ti(iyr)=etcap(iyr)+tang_dwc(iyr)+tang_ewc(iyr)+otc(iyr)
        tci(iyr)=ti(iyr)+intcap(iyr)
```

Step 13: **Total capitalized investment adjustment (*tciadj()*) is calculated.**

Note: Logical variables *eortc* (allow enhanced oil recovery tax credit always set to "no"), *tcoti* (allow tax credit on tangible investments), *tdtc* (allow tangible development tax credit), *eec* (include expense environmental costs), and *ettc* (allow environmental tangible tax credit) control the *tciadj()* calculation. Except for *eortc*, YES/NO responses for all of these logical variables are obtained from input file TAX_NAT.DAT.

```
        if(eortc) tciadj(iyr)=tciadj(iyr)+eortcr*tci(iyr)
        if(tcoti) then
         if(yr1.ge.iyr) tciadj(iyr)=tciadj(iyr)+
     &    tdtcr*(tang_dwc(iyr)+tang_ewc(iyr))+ ettcr*etcap(iyr)
        else
         if(tdtc) tciadj(iyr)=tciadj(iyr)+
     &     tdtcr*(tang_dwc(iyr)+tang_ewc(iyr))
         if(.not. eec .and. ettc) tciadj(iyr)=tciadj(iyr)+
     &    ettcr*etcap(iyr)
        endif
```

Step 14: **Total operating cost (*toc()*) and depreciation (*depr()*) are calculated.**

Note: Depreciation is calculated only if depreciable/capitalize base (*cap_base()*) is greater than zero.

```
        cap_base(iyr)=tci(iyr)-tciadj(iyr)
        ga_cap(iyr)=ga_cap_m(itech)*(ii(iyr)+ti(iyr))
        toc(iyr)=inj(iyr)+oam(iyr)+eoam(iyr)+ga_exp(iyr)+
     &   ga_cap(iyr)+stim(iyr)
        if(cap_base(iyr).gt.0) then
         do iyr1=0,7
          if(iyr+iyr1 .le. qyr)
     &    depr(iyr+iyr1)=depr(iyr+iyr1)+cap_base(iyr)*schedule(iyr1+1)
         enddo
        endif
```

Step 15: **Expensed G&G and lease acquisition (*aggla()*) is calculated.**

```
        eggla(iyr)=la(iyr)*(1-plac) + gg(iyr)*(1-pggc)
```

Step 16: **Severance tax (*sevtax()*) is calculated.**

Note: For Utah (*state=43*), no severance tax if gas production is less than 60 MCFD/well. If forgiveness of state taxes is allowed (*fsttax=.true.*, specified in input file TAX_NAT.DAT) and within the eligible years for "forgiveness of state taxes" (*iyr>yr3*), the severance tax is set to zero.

```
      sevtax(iyr)=(oilprod(iyr)*oprice(iyr)*oil_sev(istate)+
     $  oilprod(iyr)*oil_sev_p(istate)+
     &      gasprod(iyr)*gprice(iyr)*gas_sev(istate)+
     &      gasprod(iyr)*gas_sev_p(istate))*(1-royrate)
      IF (state.eq.43) THEN
      if (nwell.gt.0.0) then
      if (gasprod(iyr)*10**6/(365.0*nwell) .le. 60.0)sevtax(iyr)=0.0
      endif
      ENDIF
      if(fsttax .and. iyr .le. yr3) sevtax(iyr)=0
```

Step 17: **Depletable G&G and lease acquisition (*dggla()*), adjustment for federal tax credit (*dep_crd()*), and total G&G lease cost depletion (*tgglcd()*) are calculated.**

Note: *dggla()* and *dep_crd()* are calculated based on logical variables *ggctc* (allow G&G depletable tax credit) and *lactc* (allow lease acquisition depletable tax credit) specified in input file TAX_NAT.DAT. Total G&G lease cost depletion (*tgglcd()*) is calculated only if *dggla()* and gas production (*temp*) in the corresponding year are not zero.

```
      dggla(iyr)=gg(iyr)*pggc+la(iyr)*plac
      if(ggctc) then
        dggla(iyr)=dggla(iyr)-gg(iyr)*pggc*ggctcr
        dep_crd(iyr)=dep_crd(iyr)+gg(iyr)*pggc*ggctcr
      endif
      if(lactc) then
        dggla(iyr)=dggla(iyr)-la(iyr)*plac*lactcr
        dep_crd(iyr)=dep_crd(iyr)+la(iyr)*plac*lactcr
      endif
      temp=0.0
      do iyr1=iyr,nyr
       temp=temp+oilprod(iyr1)+gasprod(iyr1)/5.642
      enddo
      if(dggla(iyr).ne.0 .and. temp .ne.0) then
       do iyr1=iyr,nyr
        tgglcd(iyr1)=tgglcd(iyr1)+
     &     dggla(iyr)*(oilprod(iyr1)+gasprod(iyr1)/5.642)/temp
       enddo
      endif
```

Step 18: **Allowable percent depletion (*apd()*) is calculated.**

Note: *apd()* is calculated based on logical variable *nil* (allow net income limitation) which is specified in input file TAX_NAT.DAT.

```
      nilb(iyr)=netsales(iyr)-sevtax(iyr)-toc(iyr)-
     &  ii(iyr)+intcap(iyr)-eggla(iyr)-depr(iyr)
      if(nil) then
       if(nilb(iyr).gt.0) then
        apd(iyr)=min(nilb(iyr)*nill,netsales(iyr)*pdr)
       else
        apd(iyr)=0.
       endif
      else
       apd(iyr)=netsales(iyr)*pdr
      endif
```

Step 19: **Depletion (*deplet()*) is set to the higher value between total G&G lease cost depletion (*tgglcd()*) and allowable percent depletion (*apd()*).**

```
      deplet(iyr)=max(tgglcd(iyr),apd(iyr))
```

Step 20: **Net income before tax addback (*nibta()*) is calculated.**

```
      nibta(iyr)=
     &  netsales(iyr)-sevtax(iyr)-toc(iyr)-ii(iyr)+intcap(iyr)-
     &   eggla(iyr)-depr(iyr)-deplet(iyr)
```

Step 21 **Intangible drilling cost addback (*idca()*) is calculated.**

Note: *idca()* is calculated based on logical variables *tcoii* (allow tax credit on intangible investments), *cidc* (intangible drilling costs to be capitalized), *idctc* (allow intangible drilling cost tax credit), and *cidc* (intangible drilling costs to be capitalized) specified in input file TAX_NAT.DAT.

```
      if(tcoii) then
       if(yr2.ge.iyr) then
        if(cidc) then
         idca(iyr)=(1-piic)*(intang_dwc(iyr)+intang_ewc(iyr))*idctcr
        else
         idca(iyr)=(intang_dwc(iyr)+intang_ewc(iyr))*idctcr
        endif
       else
        idca(iyr)=0
       endif
      else
       if(idctc) then
        if(cidc) then
         idca(iyr)=(1-piic)*(intang_dwc(iyr)+intang_ewc(iyr))*idctcr
        else
```

```
    idca(iyr)=(intang_dwc(iyr)+intang_ewc(iyr))*idctcr
   endif
  else
   idca(iyr)=0
  endif
endif
```

Step 22 **Other intangible addback (*oia()*) is calculated.**

Note: *oia()* is calculated based on logical variables *tcoii* (allow tax credit on intangible investments), *coi* (other intangibles to be capitalized), and *oitc* (allow other intangible tax credit) specified in input file TAX_NAT.DAT.

```
if(tcoii) then
 if(yr2.ge.iyr) then
  if(coi) then
   oia(iyr)=(1-piic)*icap(iyr)*oitcr
  else
   oia(iyr)=icap(iyr)*oitcr
  endif
 else
  oia(iyr)=0
 endif
elseif(oitc) then
  if(coi) then
   oia(iyr)=(1-piic)*icap(iyr)*oitcr
  else
   oia(iyr)=icap(iyr)*oitcr
  endif
else
  oia(iyr)=0
endif
```

Step 23 **Intangible environmental addback (*iea()*) is calculated.**

Note: *iea()* is set to zero if environmental intangible tax credit is not allowed (*eitc=.false.*, specified in file TAX_NAT.DAT). *iea()* is calculated based on logical variable *ce* (environmental to be capitalized) specified in input file TAX_NAT.DAT.

```
if(eitc) then
 if(ce) then
  iea(iyr)=(1-piic)*eicap(iyr)*eitcr
 else
  iea(iyr)=eicap(iyr)*eitcr
 endif
else
 iea(iyr)=0
endif
```

Step 24 **Environmental operating cost addback (*eoca()*) is calculated.**

Note: *eoca()* is set to zero if environmental operating cost tax credit is not allowed (*eoctc=.false.*, specified in file TAX_NAT.DAT). Otherwise, it is set equal to environmental operating and maintenance cost (*eoam()*) multiplied by environmental operating cost tax credit rate (*eoctcr*).

```
if(eoctc) then
 eoca(iyr)=eoam(iyr)*eoctcr
else
 eoca(iyr)=0
endif
```

Step 25 **G&G/lease addback (*ggla()*), total intangible addback (*intadd()*), net income before taxes (*nibt()*), state income tax (*sttax()*), and federal taxable income (*fti()*) are calculated.**

Note: *ggla()* is calculated based on logical variables *ggetc* (allow tax credit for expensed G&G) and *laetc* (allow tax credit for expensed lease acquisition costs) specified in file TAX_NAT.DAT). If forgiveness of state taxes is allowed (*fsttax=.true.*, specified in input file TAX_NAT.DAT), and within the eligible years for "forgiveness of state taxes" (*yr3>=iyr*), and *nibt()* greater than zero, the state income tax (*sttax()*) is set to zero.

```
if(ggetc) then
  ggla(iyr)=ggla(iyr)+ggetcr*gg(iyr)*(1-pggc)
endif
if(laetc) then
  ggla(iyr)=ggla(iyr)+laetcr*la(iyr)*(1-plac)
endif
intadd(iyr)=idca(iyr)+oia(iyr)+iea(iyr)+eoca(iyr)
nibt(iyr)=nibta(iyr)+eortca(iyr)+intadd(iyr)+ggla(iyr)
if(fsttax .and. yr3.ge.iyr .and. nibt(iyr).gt.0) then
 sttax(iyr)=0
else
 sttax(iyr)=nibt(iyr)*strate(istate)
endif
fti(iyr)=nibt(iyr)-sttax(iyr)
```

Step 26 **Excess intangible drilling cost addback (*eidca()*), net income from oil and gas (*nifoag()*), intangible drilling cost preference for alternative minimum taxable (*idcpamt()*), unadjusted and adjusted alternative minimum taxable incomes (*uamti()* and *amti()*), ACE and ACE adjustment (*ace()* and *aceadj()*), alternative minimum taxes (*amint()*), tentative and selected federal income taxes (*tfit()* and *sfit()*), availible and usable credits for past alternative minimum taxable (*acpamt()* and *ucpamt()*), federal income tax (*fedtax()*), and balance of alternative minimum taxable paid (*bamtp()*) are calculated.**

Note: *ip* is logical variable for independent producer.

```
       eidca(iyr)=(1-smar)*(ii(iyr)-intcap(iyr))
       nifoag(iyr)=fti(iyr)+eidca(iyr)
       if(nifoag(iyr).gt.0) dpidcs(iyr)=nifoag(iyr)*ipd
       idcpamt(iyr)=eidca(iyr)-dpidcs(iyr)
       if(ip) then
        uamti(iyr)=max(fti(iyr),(1-ira)*(fti(iyr)+idcpamt(iyr)))
       else
        uamti(iyr)=fti(iyr)+idcpamt(iyr)
       endif
       if(.not. ip) then
         aceadj(iyr)=dpidcs(iyr)
         if(deplet(iyr).gt.tgglcd(iyr))
     &     aceadj(iyr)=deplet(iyr)-tgglcd(iyr)
       endif
       ace(iyr)=uamti(iyr) + aceadj(iyr)
       if(ace(iyr) .gt. uamti(iyr)) then
        amti(iyr)=uamti(iyr)+acer*(ace(iyr)-uamti(iyr))
       else
        amti(iyr)=uamti(iyr)
       endif
       amint(iyr)=amtrate*amti(iyr)
       tfit(iyr)=(nibt(iyr)-sttax(iyr))*fedrate
       if(amt) then
        sfit(iyr)=max(amint(iyr),tfit(iyr))
       else
        sfit(iyr)=tfit(iyr)
       endif
       if(iyr.eq.1) then
        acpamt(iyr)=0
       else
        acpamt(iyr)=bamtp(iyr-1)
       endif
       if(tfit(iyr) .gt. amint(iyr) .and. credamt) then
        ucpamt(iyr)=min(acpamt(iyr),tfit(iyr)-amint(iyr))
       else
        ucpamt(iyr)=0
       endif
       fedtax(iyr)=sfit(iyr)-ucpamt(iyr)
       if(iyr.eq.1) then
        bamtp(iyr)=fedtax(iyr)-tfit(iyr)
       else
        bamtp(iyr)=bamtp(iyr-1)+fedtax(iyr)-tfit(iyr)
       endif
```

Step 27 **Federal tax credits (*fedtaxc()*) is calculated.**

Note: *fedtaxc()* is calculated based on logical variables *eortc* (allow enhanced oil recovery tax credit), *tcoti* (allow tax credit on tangible investments), *ggctc* (allow G&G depletable tax credit), *ggetc* (allow tax credit for expensed G&G), *lactc* (allow lease acquisition depletable tax credit), *laetc* (allow tax credit for expensed lease acquisition costs), *tdtc* (allow tangible development tax credit), *ettc* (allow environmental tangible tax credit), *tcoii* (allow tax credit on intangible investments), *idctc* (allow intangible drilling cost tax credit), *oitc* (allow other intangible tax credit), *eitc* (allow environmental intangible tax credit), and *eoctc* (allow environmental operating cost tax credit). Except for *eortc*, YES/NO responses for all of these logical variables are obtained from input file TAX_NAT.DAT.

```
      if(eortc) fedtaxc(iyr)=fedtaxc(iyr)+
     &  eortcr*(ti(iyr)+ii(iyr)+inj(iyr))
      if(tcoti) then
       if(yr1.ge.iyr) then
         fedtaxc(iyr)=fedtaxc(iyr)+
     &      ggctcr*gg(iyr)*pggc
          fedtaxc(iyr)=fedtaxc(iyr)+
     &      lactcr*la(iyr)*plac
          fedtaxc(iyr)=fedtaxc(iyr)+
     &      tdtcr*(tang_dwc(iyr)+tang_ewc(iyr))
          fedtaxc(iyr)=fedtaxc(iyr)+
     &      ettcr*etcap(iyr)
       endif
      else
       if(ggctc) fedtaxc(iyr)=fedtaxc(iyr)+
     &  ggctcr*gg(iyr)*pggc
       if(ggetc) fedtaxc(iyr)=fedtaxc(iyr)+
     &  ggetcr*gg(iyr)*(1-pggc)
       if(lactc) fedtaxc(iyr)=fedtaxc(iyr)+
     &  lactcr*la(iyr)*plac
       if(laetc) fedtaxc(iyr)=fedtaxc(iyr)+
     &  laetcr*la(iyr)*(1-plac)
       if(tdtc) fedtaxc(iyr)=fedtaxc(iyr)+
     &  tdtcr*(tang_dwc(iyr)+tang_ewc(iyr))
       if(ettc) fedtaxc(iyr)=fedtaxc(iyr)+
     &      ettcr*etcap(iyr)
      endif
      if(tcoii) then
       if(yr2.ge.iyr) then
        fedtaxc(iyr)=fedtaxc(iyr)+
     &      idctcr*(intang_dwc(iyr)+intang_ewc(iyr))
        fedtaxc(iyr)=fedtaxc(iyr)+
     &   oitcr*icap(iyr)
       endif
      else
       if(idctc) fedtaxc(iyr)=fedtaxc(iyr)+
     &  idctcr*(intang_dwc(iyr)+intang_ewc(iyr))
       if(oitc) fedtaxc(iyr)=fedtaxc(iyr)+
     &   oitcr*icap(iyr)
      endif
      if(eitc) fedtaxc(iyr)=fedtaxc(iyr)+
     &  eitcr*eicap(iyr)
      if(eoctc) fedtaxc(iyr)=fedtaxc(iyr)+
     &  eoctcr*eoam(iyr)
```

Step 28 **Net income after taxes (*niat()*), annual after tax cash flow (*aatcf()*), discounted after tax cash flow (*datcf()*), and annual after tax cash flow (*aatcf()*) are calculated.**

```
      niat(iyr)=nibt(iyr)-sttax(iyr)-fedtax(iyr)+fedtaxc(iyr)
      aatcf(iyr)=niat(iyr)+depr(iyr)+deplet(iyr)-
     &   dggla(iyr)-intcap(iyr)-ti(iyr)-eortca(iyr)-
     &   intadd(iyr)-ggla(iyr)
      datcf(iyr)=aatcf(iyr)/((1+disc)**(iyr-1))
      if(iyr.eq.1) catcf(iyr)=datcf(iyr)
      if(iyr.gt.1) catcf(iyr)=catcf(iyr-1)+datcf(iyr)
```

Step 29: **The program control is returned back to the calling routine (sub-program CLC_MASP() or program RESVPERF) and the sub-program CASHFLOW() is ended.**

```
            Return
            End
```

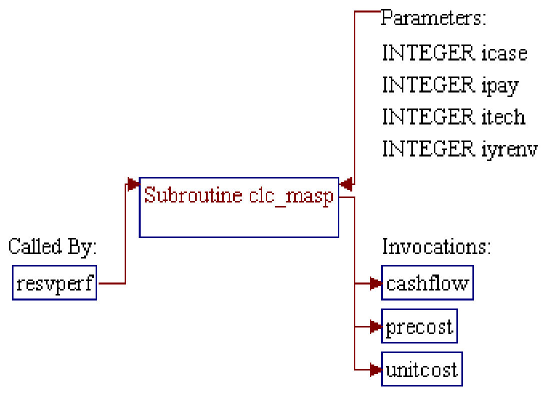
SUB-PROGRAM CLC_MASP()

LOCATION:	CLC_MASP.FOR
MAIN THEME:	This routine calculates Minimum Acceptable Supply Price (MASP) of a specified development type in a specified pay grade.

CALLS:	UNITCOST() (in file UNITCOST.FOR)
	Calculates per unit costs in $/MCF, $/Well and/or $/BBL.
	
	PRECOST() (in file PRECOST.FOR)
	Utilizes the unit cost data to create the cost streams to be fed to the cash flow routine CASHFLOW().
	
	CASHFLOW() (in file CASHFLOW.FOR)
	Performs a discounted cash flow analysis

CALLED BY:	RESVPERF (in file RESVPERF.FOR) Main program of Reservoir Performance Module.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameter of the sub-program are declared. Header ".h" files are included. Additional common block, local variables, and parameter constants are declared.**

Note: Name of the sub-program is CLC_MASP() and the parameter passed to this sub-program is as follows:

• *itech*	Technology flag (1=current, 2=advanced)
• *icase*	Development case flag (1=primary, 2=first infill, 3=second infill (not yet implemented))
• *ipay*	Pay grade number
• *iyrenv*	Number of years for environmental run (years)

```
      subroutine clc_masp(itech,icase,ipay,iyrenv)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'global.h'
      include 'cashflow.h'
      include 'costing.h'
      include 'unitcost.h'
      include 'cost.h'
      include 'field.h'
      include 'tax_nat.h'
      include 'tax_reg.h'
      include 'welldata.h'
      include 'type_out.h'
      include 'gsamvar.h'
```

Note: Additional common block, local variables, and parameter constants are declared.

```
      common/stchg/iwin_yr
      integer itech
      real*4 pricea,priceb
      real*4 npva,npvb
      real*4 newprice
      integer iter
      integer niter
      integer maxiter
      real*4 toler
      integer iyr,iyrenv,iwin_yr
      real*4 new_npv
      parameter(maxiter=50)
      parameter(toler=0.05)
```

Step 2: **Net Present Value (NPV) at $0.20/MCF gas price is calculated.**

Note: Gas price in each year is set to $0.20/MCF.

```
pricea=0.20
do iyr=1,nyr
 gprice(iyr)=pricea
enddo
```

Note: Sub-program UNITCOST() is invoked to calculate unit costs.

```
call unitcost(itech)
```

Note: Multiplier for tangible, intangible, and operating and maintenance costs are set to one.

```
do iyr=1,nyr
 tang_m(iyr)=1.0
 intang_m(iyr)=1.0
 oam_m(iyr)=1.0
enddo
```

Note: Sub-program PRECOST() is invoked to generate cost streams. Sub-program CASHFLOW() is then invoked to perform a pro-forma cash flow analysis.

```
call precost(itech,icase,iyrenv)
call cashflow(itech,1,maxcf)
```

Note: Maximum cumulative discounted after tax cash flow is obtained from array variable *catcf()* and stored in variable *npva.*

```
npva=catcf(1)
do iyr=2,nyr
 npva=max(npva,catcf(iyr))
enddo
```

Note: MASP of $0.20/MCF is returned and sub-program CLC_MASP() is terminated if the maximum cumulative discounted after tax cash flow is greater than zero (*npva>0*).

```
if(npva.gt.0) then
   masp(icase,ipay)=0.20
   return
endif
```

Step 3: **Net Present Value (NPV) at $10/MCF gas price is calculated if NPV of $0.20 gas price is less than or equal to zero.**

Note: Gas price in each year is set to $10/MCF.

```
priceb=10.00
do iyr=1,nyr
 gprice(iyr)=priceb
enddo
```

Note: Sub-program UNITCOST() is invoked to calculate unit costs.

```
call unitcost(itech)
```

Note: Multiplier for tangible, intangible, and operating and maintenance costs are set to one.

```
do iyr=1,nyr
 tang_m(iyr)=1.0
 intang_m(iyr)=1.0
 oam_m(iyr)=1.0
enddo
```

Note: Sub-program PRECOST() is invoked to generate cost streams. Sub-program CASHFLOW() is then invoked to perform a pro-forma cash flow analysis.

```
call precost(itech,icase,iyrenv)
call cashflow(itech,1,maxcf)
```

Note: Maximum cumulative discounted after tax cash flow is obtained from array variable *catcf()* and stored in variable *npvb.*

```
npvb=catcf(1)
do iyr=2,nyr
 npvb=max(npvb,catcf(iyr))
enddo
```

Note: MASP of $99/MCF is returned and sub-program CLC_MASP() is terminated if the maximum cumulative discounted after tax cash flow is negative (*npvb<0*).

```
if(npvb.lt.0) then
  masp(icase,ipay)=99.0
  return
endif
```

Step 4: **If NPV at $10/MCF gas price is positive (*npvb>=0*), an iterative procedure is performed to determine minimum gas price**

(expected between $0.2/MCF and $10/MCF) that yields positive NPV (i.e. minimum acceptable supply price).

Note: Iteration process is started by initializing iteration counter (*iter=0*) and "*continue*" flag (line number *100*) for iteration loop.

```
      iter=0
 100  continue
```

Note: New gas price (*newprice*) is estimated using linear interpolation for $0 NPV, and iteration counter (*iter*) is incremented.

```
       newprice = pricea - (npva*(pricea-priceb)/(npva-npvb))
       iter=iter+1
```

Note: Check for convergence. Return MASP of $*newprice*/MCF and terminate sub-program CLC_MASP() if convergence is achieved (gas price difference within tolerance) or if maximum number of iteration is encountered (*iter=maxiter*)

```
       if(abs(newprice/pricea-1).lt.toler .or.
     &     abs(newprice/priceb-1).lt.toler) then
        niter=iter
        masp(icase,ipay)=newprice
        return
       elseif(iter.eq.maxiter) then
        masp(icase,ipay)=newprice
        return
       endif
```

Note: Gas price in each year is set to *newprice*.

```
       do iyr=1,nyr
        gprice(iyr)=newprice
       enddo
```

Note: Sub-program UNITCOST() is invoked to calculate unit costs.

```
      call unitcost(itech)
```

Note: Multiplier for tangible, intangible, and operating and maintenance costs are set to one.

```
      do iyr=1,nyr
       tang_m(iyr)=1.0
       intang_m(iyr)=1.0
       oam_m(iyr)=1.0
       enddo
```

Note: Sub-program PRECOST() is invoked to generate cost streams. Sub-program CASHFLOW() is then invoked to perform a pro-forma cash flow analysis.

```
       call precost(itech,icase,iyrenv)
       call cashflow(itech,1,maxcf)
```

Note: Maximum cumulative discounted after tax cash flow is obtained from array variable *catcf()* and stored in variable *new_npv*.

```
        new_npv=catcf(1)
        do iyr=2,nyr
         new_npv=max(new_npv,catcf(iyr))
        enddo
```

Note: Ranges of gas price (*pricea* and *priceb*) and NPV (*npva* and *npvb*) are adjusted accordingly based on the magnitude of *new_npv*.

```
        if(new_npv.gt.0) then
         priceb=newprice
         npvb=new_npv
        else
         pricea=newprice
         npva=new_npv
        endif
```

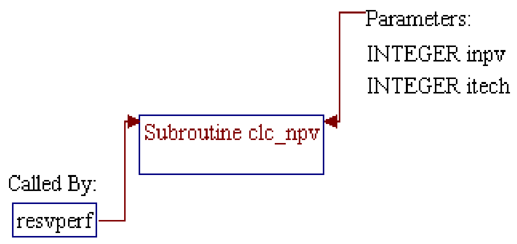
Note: The NPV calculation is repeated by looping back to *"continue"* flag (line 100).

```
        goto 100
       end
```

SUB-PROGRAM CLC_NPV()

LOCATION:	WRT_PRO.FOR
MAIN THEME:	This routine performs Net Present Value (NPV) calculations at different price and cost assumptions.
CALLS:	None
CALLED BY:	RESVPERF (in file RESVPERF.FOR) Main program of Reservoir Performance Module.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included. Local variables are declared.**

Note: Name of the sub-program is CLC_NPV() and the parameters passed to this sub-program are as follows:

• *inpv*	Flag of scenarios in cash flow calculations: 1= \$2/MCF gas price, 2=\$5/MCF gas price, 3=\$2/MCF gas price and zero drilling cost, 4=\$2/MCF gas price and zero for all other costs
• *itech*	Technology flag (1=current, 2=advanced)

```
      subroutine clc_npv(inpv,itech)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'global.h'
      include 'cashflow.h'
      include 'costing.h'
      include 'cost.h'
      include 'field.h'
      include 'tax_nat.h'
      include 'tax_reg.h'
      include 'unitcost.h'
      include 'gsamvar.h'
      include 'npv.h'
```

Note: Local variables are declared.

```
      integer inpv,itech,iyr
```

Step 2: **NPV for the specified scenario (*NPV(inpv)*) is set to cumulative discounted after tax cash flow in the final year (*catcf(nyr)*).**

```
      npv(inpv)=catcf(nyr)
```

Step 3: **Loop of year for summation of NPV (*iyr*) is initialized. Denominator of NPV equation for the current year is calculated and stored in temporary variable *temp*.**

Note: *disc* is discount rate.

```
       do iyr=1,nyr
        temp=((1+disc)**(iyr-1))
```

Step 4: **NPVs of gas production (*g_prd_npv()*) and oil production (*o_prd_npv()*) are calculated.**

Note: *gasprod()* and *oilprod()* are gas production and oil production, respectively.

```
        g_prd_npv(inpv)=g_prd_npv(inpv)+
      &   gasprod(iyr)/temp
        o_prd_npv(inpv)= o_prd_npv(inpv)+
      &   oilprod(iyr)/temp
```

Step 5: **NPV of gross sales less royalties (*gross_npv()*) is calculated.**

Note: *gprice()* and *oprice()* are gas price and oil price, respectively. *royrate* is royalty rate and *sevtax()* is severance tax.

```
        gross_npv(inpv)=gross_npv(inpv)+
      &  ((gasprod(iyr)*gprice(iyr)+
      &  oilprod(iyr)*oprice(iyr))*
      &  (1-royrate)-sevtax(iyr))/temp
```

Step 6: **NPV of total operating cost (*toc_npv()*) is calculated.**

Note: *inj()* is injectant cost (currently zero), *oam()* is operation and maintenance (O&M) cost, *eoam()* is environmental O&M cost, *ga_exp()* is G&A on expensed items, *stim()* is stimulation cost, and *recomp()* is recompletion cost.

```
        toc_npv(inpv)=toc_npv(inpv)+
      &   ( inj(iyr)+oam(iyr)+eoam(iyr)+ga_exp(iyr)+stim(iyr)+
      &    recomp(iyr) + (gasprod(iyr)*gprice(iyr)*royrate +
      &       oilprod(iyr)*oprice(iyr)*royrate) ) / temp
```

Step 7: **NPV of intangible investment (*int_npv()*) is calculated.**

Note: *ii()* is intangible investment, *intang_dwc()* is intangible development cost, *intang_ewc()* is intangible exploratory cost, *ga_cap_m()* is G&A capital multiplier, *icap()* is intangible capital, and *eicap()* is environmental intangible capital cost.

```
       int_npv(inpv)=int_npv(inpv)+
     &    (ii(iyr)-(intang_dwc(iyr)+intang_ewc(iyr)))/temp +
     &    (ga_cap_m(itech)*(icap(iyr)+eicap(iyr)))/temp
```

Step 8: **NPV of tangible investment excluding drilling (*tan_npv()*) is calculated.**

Note: *ti()* is tangible investment, *tang_dwc()* is tangible development cost, *tang_ewc()* is tangible exploratory cost, *etcap()* is environmental tangible capital cost, and *otc()* is other tangible capital.

```
       tan_npv(inpv)=tan_npv(inpv)+
     &    (ti(iyr)-(tang_dwc(iyr)+tang_ewc(iyr)))/temp +
     &    (ga_cap_m(itech)*(etcap(iyr)+otc(iyr)))/temp
```

Step 9: **NPV of development well cost (*dwc_npv()*) is calculated.**

```
       dwc_npv(inpv)=
     &    dwc_npv(inpv)+(tang_dwc(iyr)+intang_dwc(iyr))/temp +
     &    (ga_cap_m(itech)*(intang_dwc(iyr)+tang_dwc(iyr)))/temp
```

Step 10: **NPV of exploratory well cost (*ewc_npv()*) is calculated.**

```
       ewc_npv(inpv)=
     &    ewc_npv(inpv)+(tang_ewc(iyr)+intang_ewc(iyr))/temp +
     &    (ga_cap_m(itech)*(intang_ewc(iyr)+tang_ewc(iyr)))/temp
```

Step 11: **NPV of state and federal taxes (*tax_npv()*) is calculated.**

Note: *fedtax()* is federal income tax and *sttax()* is state income tax.

```
       tax_npv(inpv)=tax_npv(inpv)+(fedtax(iyr)+sttax(iyr)+sevtax(iyr))
     $   /temp
```

Step 12: **NPV of depletable G&G/lease acquisition (*depggla_npv()*) is calculated.**

Note: *dggla()* is depletable G&G and lease acquisition cost.

```
       depggla_npv(inpv)=depggla_npv(inpv)+dggla(iyr)/temp
```

Step 13: **NPV of expensed G&G/lease acquisition (*expggla_npv()*) is calculated.**

Note: *eggla()* is expensed G&G and lease acquisition cost.

```
      expggla_npv(inpv)=expggla_npv(inpv)+eggla(iyr)/temp
```

Step 14: **NPV of federal tax credits (*credit_npv()*) is calculated.**

Note: *fedtaxc()* is federal tax credit.

```
      credit_npv(inpv)=credit_npv(inpv)+fedtaxc(iyr)/temp
```

Step 15: **NPV of total cost of the reservoir (for current pay grade and development type) for current scenario (*totalcst()*) is calculated.**

Note: *toc()* is total operating cost, *la()* is lease acquisition cost, and *gg()* is G&G cost.

```
      totalcst(inpv)=totalcst(inpv)+toc(iyr)+ii(iyr)+
     &   ti(iyr)+la(iyr)+gg(iyr)
```

Step 16: **Loop of year for summation of NPV is closed.**

```
     enddo
```

Step 17: **NPV of state and NPV of federal taxes (*tax_npv()*) are limited between MM$ -999.999 and 9999.999.**

```
        if(tax_npv(inpv).gt.9999.999)tax_npv(inpv)=9999.999
        if(tax_npv(inpv).lt.-999.999)tax_npv(inpv)=-999.999
```

Step 18: **NPV of total invesments (*tot_inv()*) is calculated.**

```
       tot_inv(inpv)=int_npv(inpv)+tan_npv(inpv)+
     &  dwc_npv(inpv)+ewc_npv(inpv)+depggla_npv(inpv)+expggla_npv(inpv)
```

Step 19: **NPV of drilling costs (*drl_inv()*) is calculated.**

```
      drl_inv(inpv)=dwc_npv(inpv)+ewc_npv(inpv)
```

Step 20: **The program control is returned back to the calling routine (program RESVPERF()) and the sub-program CLC_NPV() is ended.**

```
      return
      end
```

SUB-PROGRAM PRECOST()

LOCATION: PRECOST.FOR

MAIN THEME: This routine utilizes the unit cost data to create the cost streams to be fed to the cash flow routine CASHFLOW().

CALLS: INITCOST (in file INITIAL.FOR)
Initializes costing variables as declared in header file COSTING.H.

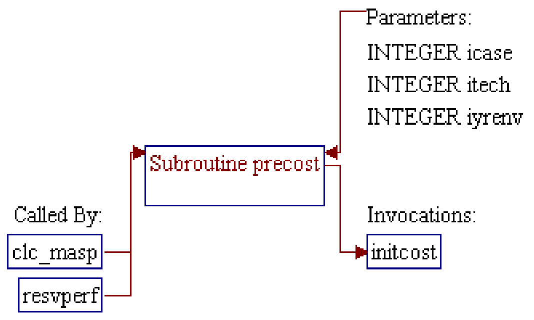
CALLED BY: RESVPERF (in file RESVPERF.FOR)
Main program of Reservoir Performance Module.

CLC_MASP() (in file CLC_MASP.FOR)
Calculates Minimum Acceptable Supply Price (MASP) of a specified development type in a specified pay grade

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameter of the sub-program are declared. Header ".h" files are included. Additional common block and local variables are declared.**

Note: Name of the sub-program is PRECOST() and the parameter passed to this sub-program is as follows:

• *itech*	Technology flag (1=current, 2=advanced)
• *icase*	Development case flag (1=primary, 2=first infill, 3=second infill (not yet implemented))
• *iyrenv*	Number of years for environmental run (years)

```
      subroutine precost(itech,icase,iyrenv)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'global.h'
      include 'cost.h'
      include 'field.h'
      include 'costing.h'
      include 'tax_nat.h'
      include 'tax_reg.h'
      include 'unitcost.h'
      include 'welldata.h'
      include 'type_out.h'
      include 'gsamvar.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
```

Note: Additional common block and local variables are declared.

```
       common /stchg/ iwin_yr
       integer iyr,itech,icase
       real*4 fxoam(qyr)
       real*4 voam(qyr)
       real*4 h2ovoam(qyr)
       real*4 faccost(qyr)
       real*4 oam_comp(qyr)
       integer comp_yr
       integer nyr_cash
       integer winyr,iyrenv
```

Step 2: **Number of years for cash flow (*nyr_cash*) is set equal to number of years in type curve run (*nyr*).**

```
        nyr_cash=nyr
```

Step 3: **Window year (*winyr*) is set to be the minimum between number of years in type curve plus one (*nyr+1*), calculated window year (*iwin_yr*), and maximum allowable number of years for RP run (*qyr*).**

Note: Window year is the number of years for which the total flow rate of the reservoir (from the three pay grades) remains constant.

```
        winyr=min(nyr+1,iwin_yr,qyr)
```

Step 4: **Fixed operating and maintenance (O&M) cost (*fxoam()*), surface gas O&M cost (*voam()*), surface water O&M cost (*h2ovoam()*), facilities cost (*faccost()*), and compressor O&M cost (*oam_comp()*) in each year are initialized to zero.**

```
        do iyr=1,qyr
         fxoam(iyr)=0.0
         voam(iyr)=0.0
         h2ovoam(iyr)=0.0
         faccost(iyr)=0.0
         oam_comp(iyr)=0.0
        enddo
```

Step 5: **Sub-program INITCOST is invoked to initialized other costing variables.**

Note: These variables are declared in header file COSTING.H.

```
        call initcost
```

Step 6: **The year in which compression will commence (*comp_yr*) is set equal to the year when wellhead pressure (*type_pwhp()*) drops below the minimum allowable wellhead pressure (*premin*).**

```
          comp_yr=-1
          do iyr=1,nyr_cash
           if(type_pwhp(1,1,iyr).lt.premin) then
            comp_yr=iyr
            goto 555
           endif
          enddo
   555   continue
```

Step 7: **Development well cost (*dwc()*) and stimulation cost (*stim()*) are calculated.**

Note: The following code calculates development well cost and stimulation cost for primary wells without refrac. *dwc_w* is development well unit cost, *prob_dry()* is probability of dry hole, *pdry_dev()* is percentage of dry hole cost as development cost, *nwell* is number of wells, *stim_w* is stimulation unit cost, and *intang_m* is intangible multiplier.

```
          dwc(1)=dwc(1)+((dwc_w)+(prob_dry(itech)*dwc_w*
     &      pdry_dev(itech)))*nwell
          stim(1)=stim(1)+
     &      stim_w*nwell*intang_m(1)
```

Note: For refrac (*icase=2*), the wells are hydraulically fractured in year *winyr*. The following code calculates stimulation cost in *winyr* (*stim(winyr)*).

```
      if(icase.eq.2) then !refrac wells
        if(winyr.gt.0.and.winyr.le.nyr) then
          stim(winyr)=stim(winyr)+
     &      stim_w*nwell*intang_m(winyr)
        endif
      endif
```

Note: For infill well case (*icase=3*), new wells are drilled in year *winyr* and hydraulic fracturing is performed to these new wells. The following code calculates development well cost and stimulation cost in *winyr* (*dwc(winyr)* and *stim(winyr)*).

```
       if(icase.eq.3) then !infill wells
        if(winyr.gt.0.and.winyr.lt.nyr) then
          dwc(winyr)=dwc(winyr)+((dwc_w)+
     &     (prob_dry(itech)*dwc_w*
     &      pdry_dev(itech)))*nwell
           stim(winyr)=stim(winyr)+
     &     stim_w*2*nwell*intang_m(winyr)
        endif
       endif
```

Step 8: **If compression is commenced during cash flow years (*comp_yr>0*), the compressor cost in year *comp_yr* (*comp(comp_yr)*) and compressor operating and maintenance**

costs from year *comp_yr* to *nyr_cash* (*oam_comp()*) are calculated.

Note: *comp_w* is compressor unit cost, *tang_m()* is tangible multiplier, *gasprod()* is gas production, and *comp_oam* is compressor O&M unit cost.

```
if(comp_yr.gt.0) then
 comp(comp_yr)=comp_w*nwell*tang_m(comp_yr)
  do iyr=comp_yr,nyr_cash
   oam_comp(iyr)=gasprod(iyr)*comp_oam
  enddo
endif
```

Step 9: **Exploratory well cost (*ewc()*), G&G cost (*gg()*), and facilities cost (*faccost()*) are calculated.**

Note: The following code calculates exploratory well cost (assumed zero in RP module), G&G cost (defined as fraction of exploratory well cost), and facilities cost for primary wells without refrac. *ewc_w* is exploratory well unit cost, *gg_fac()* is G&G factor, and *fac_w()* is facilities unit cost.

```
ewc(1)=ewc_w*0.0
gg(1)=ewc(1)*intang_m(1)*gg_fac(itech)
ewc(1)=ewc(1)-gg(1)
faccost(1) = fac_w*nwell
```

Step 10: **Fixed operating and maintenance cost in each year (*fxoam()*) are calculated.**

Note: *fxoam_w* is fixd O&M unit cost.

```
do iyr=1,nyr_cash
  fxoam(iyr)=nwell*fxoam_w
enddo
```

Step 11: **Surface gas operating and maintenance cost (*voam()*), surface water operating and maintenance cost (*h2ovoam()*), total operating and maintenance cost (*oam()*), total environmental operating and maintenance cost (*eoam()*), other tangible cost (otc()), and other intangible cost (*icap()*) in each year are calculated.**

Note: *voam_g* is surface gas O&M unit cost, *h2ooam_w* is surface water O&M unit cost, *oam_m()* is O&M multiplier, *env_oam_w* is environmental water O&M unit cost, *env_oam_g* is environmental gas O&M unit cost, and *fac_tan()* fraction of tangible cost.

```
      DO 96 iyr=1,nyr_cash
       voam(iyr)=gasprod(iyr)*voam_g
       h2ovoam(iyr)=h2oprod(iyr)*h2ooam_w
       oam(iyr)=
     &   (fxoam(iyr)+voam(iyr)+h2ovoam(iyr)+oam_comp(iyr))*oam_m(iyr)
       eoam(iyr)=
     &  (h2oprod(iyr)*env_oam_w+gasprod(iyr)*env_oam_g)*oam_m(iyr)
       otc(iyr)=faccost(iyr)*fac_tan(itech)*tang_m(iyr)+comp(iyr)
       icap(iyr)=faccost(iyr)*(1-fac_tan(itech))*intang_m(iyr)
 96   CONTINUE
```

Step 12: **Environmental tangible capital cost (*etcap()*), environmental intangible capital cost (*eicap()*), and total environmental operating and maintenance cost (*eoam()*) are calculated.**

Note: *env_cap_w* is environmental capital unit cost.

```
          etcap(1)= fac_tan(itech)*env_cap_w*tang_m(1)
          eicap(1)=(1-fac_tan(itech))*env_cap_w*tang_m(1)
```

Step 13: **For the case when environmental RP run is requested in input file RUNSET.DAT which is indicated by *0<iyrenv<=nyr*, environmental tangible capital cost (*etcap()*), environmental intangible capital cost (*eicap()*), and total environmental operating and maintenance cost (*eoam()*) are modified accordingly.**

Note: *etcap()*, *eicap()*, and *eoam()* for the case when *iyrenv<=1* (environmental RP run starts during the first simulation year) are modified. *envni* is intangible environmental new well unit cost, *envnt* is tangible environmental new well unit cost, *env_oam_n* new wells environmental O&M unit cost.

```
      If (iyrenv.gt.0.and.iyrenv.le.nyr) Then
        IF (iyrenv.le.1) THEN
         eicap(1) = eicap(1) + envni*nwell*(1.0 + prob_dry(itech))
         etcap(1) = etcap(1) + envnt*nwell
         Do iyr = 1, nyr_cash
          eoam(iyr) = eoam(iyr) + env_oam_n*nwell
         Enddo
```

Note: *etcap()*, *eicap()*, and *eoam()* for *iyrenv>1* are modified. *envei* is intangible environmental unit cost, *envet* is tangible environmental unit cost, *env_oam_l* environmental O&M unit cost.

```
  ELSE
   eicap(iyrenv) = eicap(iyrenv) + envei*nwell
   etcap(iyrenv) = etcap(iyrenv) + envet*nwell
   Do iyr = iyrenv,nyr_cash
    eoam(iyr) = eoam(iyr) + env_oam_l*nwell
   Enddo
  endif
endif
```

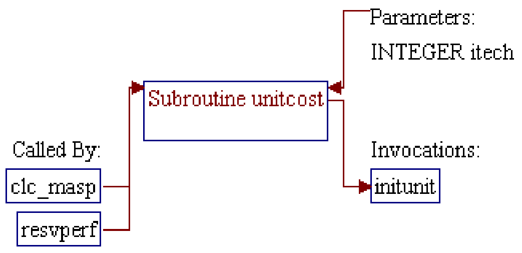
Step 14: **The program control is returned back to the calling routine (sub-program CLC_MASP() or program RESVPERF) and the sub-program PRECOST() is ended.**

```
  Return
  End
```

SUB-PROGRAM UNITCOST()

LOCATION:	UNITCOST.FOR
MAIN THEME:	This routine calculates per unit costs in $/MCF, $/Well and/or $/BBL.
CALLS:	INITUNIT (in file INITIAL.FOR) Initializes cash flow variables as declared in header file UNITCOST.H.
CALLED BY:	RESVPERF (in file RESVPERF.FOR) Main program of Reservoir Performance Module. CLC_MASP() (in file CLC_MASP.FOR) Calculates Minimum Acceptable Supply Price (MASP) of a specified development type in a specified pay grade.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameter of the sub-program are declared. Header ".h" files are included. Local variables are declared.**

Note: Name of the sub-program is UNITCOST() and the parameter passed to this sub-program is as follows:

- *itech* Technology flag (1=current, 2=advanced)

```
      subroutine unitcost(itech)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'global.h'
      include 'field.h'
      include 'cost.h'
      include 'unitcost.h'
      include 'tax_nat.h'
      include 'gsamvar.h'
      include 'welldata.h'
      include 'type_out.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
```

Note: Local variables are declared.

```
      integer istep,findstep,iyr,itech,istep1
```

Step 2: **Sub-program INITUNIT is invoked to initialize unit cost variables declared in header file UNITCOST.H.**

```
      call initunit
```

Step 3: **Stimulation cost (*stim_w*) for vertical well (*jtyp()=0*) or horizontal well (*jtyp()=1*) is calculated.**

Note: *avdep* is depth to the center of the reservoir, *halfln()* is fracture half length, *netpay* is net pay thickness, *stim_fac()* is development well cost for stimulation length, and *1/1E6* is conversion factor from $ to MM$. Stimulation cost is $20,000 + $2.5/foot + fracturing cost. Variable *stimfac* is stimulation efficiency. For horizontal wells no fracturing is assumed.

```
        if (jtyp(1,1).ge.1) then
         stim_w = (20000 + 2.5*avdep)/1.e6
        else
         stim_w=(20000+2.5*avdep+(1.5*halfln(2,1)*netpay))/stimfac(itech)
      % /1.e6
        endif
```

Step 4: **Compressor cost (*comp_w*) is calculated and compressor operating and maintenance cost (*comp_oam*) is assigned.**

Note: Pressure *pressin* is set to minimum wellhead pressure *premin*. In case *premin* value is not available it is set to 250 psia. *peakrate* is peak production rate, *cost_bhp()* is cost of compressor installation, *comp_vc()* is compressor operating and maintenance cost obtained from input file COST.DAT.

```
       prssin=premin
       if(prssin.le.0.0) prssin=250.0
       comp_w=(22*1000/prssin*peakrate/1000)/1.e6
       comp_w  = comp_w*cost_bhp(itech)
       comp_oam=comp_vc(itech)
```

Step 5: **Development well unit cost (*dwc_w*), and exploratory well unit cost (*ewc_w*) are calculated. Environmental costs are calculated/assigned.**

Note: Location of data for development well cost in input file COST.DAT (*ireg*) is searched. Sequential searching technique is performed until the region number in the first column of the data (*dwc_reg()*) matches the GSAM supply region given in variable *gsamsr*. Number of data for development well cost calculation (excluding the default data) is *ndwcreg()*. If no match is found (*ireg* is equal to *ndwcreg()*), default data specified in location number *qreg+1* is utilized.

```
       do ireg=1,ndwcreg(itech)
        if (gsamsr.eq.dwc_reg(itech,ireg)) goto 39234
       enddo
       ireg=qreg+1
 39234 continue
```

Note: Location of data for environmental costs in input file COST.DAT (*iregst*) is searched using the same technique as in the development well cost data. The environmental costs data in input file COST.DAT can be entered based on GSAM supply region (number of data is at most 40) or based on State/District (number of data greater than 40). In the following code, number of data (*newcreg()*) is used to identify whether the data is based on GSAM supply region or State. The searching algorithm will use GSAM supply region (*gsamsr*) if *newcreg()<=40*, or State/District code (*state*) if *newcreg()<=40*. The searching procedure is stopped when the ID number in the first column of the data (*ewc_reg()*) matches the value in *gsamsr* or *state.* If no match is found (*iregst* is equal to *newcreg()*), default data specified in location number *newcreg()+1* is utilized.

```
      if(newcreg(itech).gt.40) then
      do iregst=1,newcreg(itech)
       if(state.eq.ewc_reg(itech,iregst)) goto 39235
      enddo
      iregst=newcreg(itech)+1
      goto 39235
      endif
      do iregst=1,newcreg(itech)
       if(gsamsr.eq.ewc_reg(itech,iregst)) goto 39235
      enddo
      iregst=newcreg(itech)+1
```

Note: Development well unit cost (*dwc_w*) is calculated using a 4-order polynomial equation that fits the historical cost versus depth data from the 1997 JAS Survey. Four coefficients given in the development well cost data designated by pointer *ireg* (*dwck(), dwcx(), dwcxx(),* and *dwcxxx()*) are utilized. The calculated development well unit cost is then divided by drilling cost factor (*dcstf()*) which is the last entry in the data in line *ireg*. *1/1000* is a conversion factor from M$ to MM$.

```
39235  dwc_w=dwck(itech,ireg)+dwcx(itech,ireg)*avdep+
     &   dwcxx(itech,ireg)*avdep**2+dwcxxx(itech,ireg)*avdep**3
       dwc_w=dwc_w*dcstf(itech,ireg)/1000.0
```

Note: Exploratory well unit cost (*ewc_w*) is equal to sum of development well cost (*dwc_w*) and stimulation cost (*stim_w*) multiplied by exploratory well cost factor (*ewc_fac()*) obtained from input file COST.DAT.

```
ewc_w=ewc.fac(itech)*(dwc.w + stim.w)
```

Note: Environmental well cost (*envw*) is the total of new well environmental tangible and intangible capital costs (*env_nt()* and *env_ni()*). These tangible and intangible costs are then stored into variables *envnt* and *envni*, respectively. Existing well environmental tangible and intangible capital costs (*env_nt()* and *env_ni()*) are stored into variables *envnt* and *envni*, respectively. *1/1000* in all the environmental costs is a conversion factor from M$ to MM$.

```
        envw=env_nt(itech,iregst)+env_ni(itech,iregst)
        envw=envw/1000.
        envni=env_ni(itech,iregst)/1000.
        envnt=env_nt(itech,iregst)/1000.
        envei=env_ei(itech,iregst)/1000.
        envet=env_et(itech,iregst)/1000.
```

Step 6: **Facilities well unit cost (*fac_w*) is calculated.**

Note: Location of data for facilities well cost in input file COST.DAT (*ireg*) is searched. Sequential searching technique is performed until the region number (*faci_reg()*) matches the GSAM supply region given in variable *gsamsr*. Number of data for development well cost calculation (excluding the default data) is *nreg_faci().* If no match is found (*ireg* is equal to *nreg_faci()*), default data specified in location number *qreg+1* is utilized.

```
         Do ireg=1,nreg_faci(itech)
          if (gsamsr.eq.faci_reg(itech,ireg)) goto 49236
         enddo
         ireg=qreg+1
 49236   continue
```

Note: The facilities well cost calculation in RP Module is designed so that several number of steps for different depths (if desired) can be implemented. The following code searches for the location of data for the associated step in input file COST.DAT (*istep1*). The searching procedure is performed sequentially until the average reservoir depth (*avdep*) is greater or equal to the depth data (*faci_max()*). Note that the depth data in file COST.DAT is entered in increasing order. If no match is found, data in the first step (*istep1=1*) is utilized. Prior to performing the searching procedure, *avdep* is first compared with the last entry of depth data (*faci_max(fac_n(),...,...)*) where *fac_n()* is the number of steps. If *avdep* is greater or equal to *faci_max(fac_n(),...,...)* the *fac_n()* is used as the step pointer (*istep1*).

```
        if (avdep.ge.faci_max(fac_n(itech,ireg),itech,ireg)) then
```

```
          istep1 = fac_n(itech,ireg)
          goto 221
        else
         do istep1=2,fac_n(itech,ireg)
          if (avdep.ge.faci_max(istep1-1,itech,ireg)) GoTo 221
         enddo
         istep1 = 1
        endif
221     continue
```

Note: The facilities well cost (*fac_w*) is calculated. *faci_k()* is the facility cost constant factor ($/well), *faci_s()* is facility cost slope factor ($/well/MCFD), *peakrate* is peak production rate, and *1/1E6* is conversion factor from $ to MM$.

```
         fac_w = (faci_k(istep1,itech,ireg) +
     &    faci_s(istep1,itech,ireg)*peakrate)/1e6
```

Step 7: **Fixed operating and maintenance well cost (*fxoam_w*) is calculated.**

Note: Similar to the facilities well cost, several number of regions can be entered for fixed O&M cost data, and several number of steps for different depths can be implemented for fixed O&M cost calculation. The following code searches for the region number (*ireg*) and the location of data for the associated step (*istep1*) using similar technique in Step 6 except that the direction in searching algorithm for *istep1* starts from the depth entry before the last step to the beginning of the step. *nreg_fx()* is number of regions, *fxoam_reg()* is region number data, *fxoam_max()* is depth data, and *fxoam_n()* is number of steps.

```
       do ireg=1,nreg_fx(itech)
        if(gsamsr.eq.fxoam_reg(itech,ireg)) goto 39236
       enddo
       ireg=qreg+1
39236  continue
       if(avdep.ge.fxoam_max(fxoam_n(itech,ireg),itech,ireg)) then
         istep=fxoam_n(itech,ireg)
         goto 121
       else
        do istep=fxoam_n(itech,ireg)-1,1,-1
         if(avdep.ge.fxoam_max(istep,itech,ireg)) goto 121
        enddo
        istep=1
       endif
  121  continue
```

Note: The fixed O&M well cost (*fxoam_w*) is calculated. *fxoam_k()* is the fixed O&M cost constant factor ($/well), *fxoam_s()* is fixed O&M cost slope factor ($/well-ft), *avdep* is reservoir depth, and *1/1E6* is conversion factor from $ to MM$. Two equations are utilized to avoid error due to accessing out of bound array *fxoam_max(istep-1,...,...)* in the case of *istep=1* (reservoir depth is less than the first entry of depth array).

```
       if(istep.eq.1) then
         fxoam_w = (fxoam_k(istep,itech,ireg) +
     &   fxoam_s(istep,itech,ireg)*avdep)/1e6
       else
         fxoam_w = (fxoam_k(istep,itech,ireg) +
     &   fxoam_s(istep,itech,ireg)*
     &   (avdep-fxoam_max(istep-1,itech,ireg)))/1e6
       endif
```

Step 8: **Surface operating and maintenance water cost (*h2ooam_w*) is set equal to the value specified in input file COST.DAT (*oam_h2o()*).**

```
       h2ooam_w=oam_h2o(itech)
```

Step 9: **Variable operating and maintenance gas cost (*voam_g*) is set equal to the sum of operating and maintenance gas cost (*oam_gas()*) and operating and maintenance cost per 1000 feet of well depth (*oam_inc()*avdep/1000*).**

Note: *oam_inc()* is incremental operating and maintenance cost per 1000 feet, *avdep* is reservoir depth, and *1/1000* is used to calculate the incremental factor.

```
       voam_g=oam_gas(itech)+oam_inc(itech)*avdep/1000
```

Step 10: **Lease bonus fraction (*lbc_frac*) which is a fraction of total gas revenues is set equal to lease bonus cost factor specified in input file COST.DAT (*lbc_fac()*).**

```
       lbc_frac=lbc_fac(itech)
```

Step 11: **Environmental capital costs for existing and new wells (*env_cap_w* and *env_cap_n*) are set equal to the facilities well unit cost (*fac_w*) multiplied with environmental capital cost multiplier (*eccm()*) specified in input file COST.DAT.**

```
        env_cap_w=eccm(itech)*fac_w
        env_cap_n=eccm(itech)*fac_w
```

Step 12: **Environmental operating and maintenance costs for gas and water (*env_oam_g* and *env_oam_w*) are set equal to user specified data in input file COST.DAT (*env_g* and *env_w*).**

```
        env_oam_g=env_g(itech,iregst)
        env_oam_w=env_w(itech,iregst)
```

Step 13: **Environmental operating and maintenance costs for existing and new wells (*env_oam_l* and *env_oam_n*) are set equal to the user specified data in input file COST.DAT (*env_ee* and *env_ne*).**

```
        env_oam_l=env_ee(itech,iregst)/1e3
        env_oam_n=env_ne(itech,iregst)/1e3
```

Step 14: **Tangible cost multiplier (*tang_m()*), intangible cost multiplier (*intang_m()*), and operating and maintenance multiplier (*oam_m()*) for gas in each year are calculated.**

Note: *gprice()* is gas price ($/MCF).

```
       do iyr=1,qyr
        tang_m(iyr)=1+0.3*(gprice(iyr)-2.)/2.
        intang_m(iyr)=1+0.4*(gprice(iyr)-2.)/2.
        oam_m(iyr)=1+0.2*(gprice(iyr)-2.)/2.
       enddo
```

Step 15: **The program control is returned back to the calling routine (sub-program CLC_MASP(), or program RESVPERF) and the sub-program UNITCOST() is ended.**

```
       return
       end
```

SUB-PROGRAM DATOUT()

LOCATION: MODULE6D.FOR

MAIN THEME: This routine prints out results to type curve output files (.TCO files) as requested in input file REGIONS.DAT.

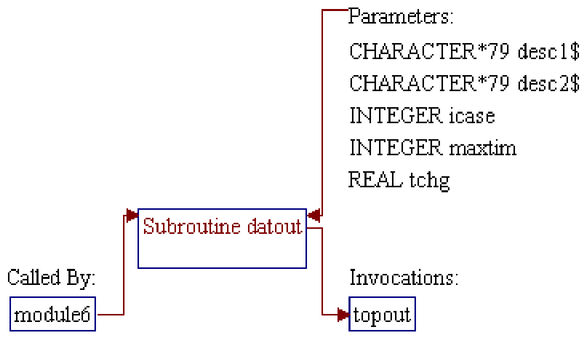
CALLS: TOPOUT() (in file MODULE6C.FOR)
Prints out two header/description lines to type curve output file (.TCO).

CALLED BY: MODULE6() (in file MODULE6A.FOR)
Controls the type curve modules in generating type curve data.

READS: None

CREATES: [GSAM].TCO
(Type curve output files)

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included. Local variables are declared.**

Note: Name of the sub-program is DATOUT() and the parameters passed to this sub-program are as follows:

• *Desc1$*	First line of description
• *Desc2$*	Second line of description
• *MaxTim*	Maximum number of time steps
• *ICase*	Case number: 1=primary, 2=automatic refrac, 3=automatic infill one time, 4=automatic infill twice (not yet implemented).
• *TChg*	Time at which automatic change in development type occurs (automatic infill or refrac)

```
      SUBROUTINE DatOut (Desc1$, Desc2$, MaxTim, ICase, TChg)
```

Note: Header .h files which declare global variables and common blocks are included.

```
    include 'dimen.h'
    include 'welldata.h'
    include 'type_out.h'
    include 'type5.h'
    include 'type3.h'
    include 'type4.h'
    include 'type8.h'
    include 'type10.h'
```

Note: Local variables are declared.

```
      Dimension Wells(3,3)
      Character*79 Desc1$, Desc2$
```

Step 2: **Number of wells (*Wells()*) and total number of wells (*SumWel*) are calculated.**

```
      SumWel = 0.
      Do I=1,3
              Wells(I,1) = Area(I) / WSpace(I)
              If ((ICase .eq. 1) .or. (ICase .eq. 2)) then
                      Wells(I,2) = 0.
                      Wells(I,3) = 0.
              Else
                      Wells(I,2) = Wells(I,1)
                      Wells(I,3) = 0.
                      If (ICase .eq. 4) Wells(I,3) = Wells(I,1) * 2.
```

```
                 End If
                 SumWel = SumWel + Wells(I,1) + Wells(I,2) + Wells(I,3)
```

Step 3: **Wells are shut in, flow rates and pressures are readjusted if total gas flow rate is too low.**

Note: If total gas flow rate is less than 10 MCFD/well, water influx (*WtrInf()*) is set to zero, average reservoir pressure (*PreAvg()*) is set to initial pressure (*Pinit()*) for the first time step or set to previous value for time step greater than 1.

```
                 Do ITime = 1, MaxTim
                    QgT = Qg(I,1,ITime)+Qg(I,2,ITime)+2.*Qg(I,3,ITime)
                    If (QgT.lt.10.) then
                       WtrInf(I,ITime) = 0.
                       PreAvg(I,ITime) = Pinit(I)
                       If (ITime.gt.1) PreAvg(I,ITime)=PreAvg(I,ITime-1)
                    End If
```

Note: Furthermore, if gas flow rates from both primary and infill wells (*Qg(I,1,ITime)* and *Qg(I,1,ITime)*) are less than 1 MCFD/well, the flow rates and pressures are set to zeros. Cumulative gas production (*CumGas()*) is set to zero for the first time step or set to previous value for time step greater than 1.

```
                    Do J = 1, 3
                       If ((Qg(I,1,ITime).lt.1.).and.
     +                    (Qg(I,2,ITime).lt.1.)) then
                          CAOF(I,J,ITime)   = 0.
                          Prbh(I,J,ITime)   = 0.
                          Prwh(I,J,ITime)   = 0.
                          Qw  (I,J,ITime)   = 0.
                          Qg  (I,J,ITime)   = 0.
                          CumGas(I,J,ITime) = 0.
                          If (ITime.gt.1)
     +                    CumGas(I,J,ITime)=Max(CumGas(I,J,ITime),
     +                                          CumGas(I,J,ITime-1))
                       End If
                    End Do
                 End Do
         End Do
```

Step 4: **Header lines are printed. Sub-program TOPOUT() is invoked to print two description lines.**

Note: First, a carriage return (format line 999) and 20 blank lines are printed followed by two description lines from sub-program TOPOUT(). A divider line (format line 100) and header lines (format line 101) are then printed.

```
        Write (55,999)
        Do I = 1, 20
               Write (55,*)
        End Do
        Call TOPOUT (ICase, Desc1$, Desc2$)
        Write (55,*)
        Write (55,100)
        Write (55,101)
```

Step 5: **Pay grade level cumulative gas production *(GrossG)*, original gas in place *(Orig)*, and percent recovery *(Recvry)* are calculated and printed. Total recovery *(TotRec)*, total gas in place *(TotGIP)*, and total percent recovery *(Recvry)* are also calculated and printed.**

```
        TotGIP = 0.
        TotRec = 0.
        Do I = 1, 3
               Orig   = OGIP1(I) * Wells(I,1) / 1000.
               GrossG = 0.
               Do J = 1, 3
                      GrossG = GrossG + CumGas(I,J,MaxTim) *
     +                        Wells(I,J) / 1000.
               End Do
               Recvry = GrossG / Orig * 100.
               TotGIP = TotGIP + Orig
               TotRec = TotRec + GrossG
               Write (55,102) I, GrossG, Orig, Recvry
        End Do
        Recvry = TotRec / TotGIP * 100.
        Write (55,103) TotRec, TotGIP, Recvry
```

Step 6: **Time of development type change (*TChg*) is printed if the change (automatic infill or refrac) take place during the course of the simulation (*TChg>0*).**

```
        If (TChg .gt. 0.) then
               If (ICase .eq. 1) then
                Write (55,104) TChg
               Endif
               If (ICase .eq. 2) Write (55,105) TChg
               If (ICase .eq. 3) Write (55,106) TChg
               If (ICase .eq. 4) Write (55,107) TChg
        End If
```

Step 7: **Number of wells in each pay grade for each development type (*Wells()*),total number of well (*SumWel*), gas flow rate from each pay grade (*Qg()*), and total gas flow rate (*QTotal*) are printed.**

Note: Prior to printing this information, sub-program TOPOUT() is reinvoked to print description lines.

```
         Write (55,999)
         Call TOPOUT (ICase, Desc1$, Desc2$)
         Write (55,110)
         Write (55,140) ((Wells(I,J),J=1,3),I=1,3), SumWel
         Write (55,151)
         Do K = 1, MaxTim
                 QTotal = 0.
                 Do I=1,3
                         Do J=1,3
                                 QTotal = QTotal + Qg(I,J,K) * Wells(I,J)
                         End Do
                 End Do
                 Write (55,201) Time(K),((Qg(I,J,K),J=1,3),I=1,3),QTotal
                 If (Mod(K,5) .eq. 0) Write(55,160)
         End Do
```

Step 8: **Cumulative gas production *(CumGas())* at each time step is printed.**

Note: Prior to printing this information, number of wells in each pay grade for each development type, total number of wells, and two description lines from sub-program TOPOUT() are printed.

```
         Write (55,999)
         Call TOPOUT (ICase, Desc1$, Desc2$)
         Write (55,112)
         Write (55,140) ((Wells(I,J),J=1,3),I=1,3), SumWel
         Write (55,152)
         Do K = 1, MaxTim
             Cum = 0.
             Do I=1,3
                 Do J=1,3
                     If (K.gt.1) CumGas(I,J,K)=Max(CumGas(I,J,K),
      +                                           CumGas(I,J,K-1))
                     Cum = Cum + CumGas(I,J,K) * Wells(I,J)
                 End Do
             End Do
             Write (55,201) Time(K), ((CumGas(I,J,K)/1000., J=1,3),
      +                    I=1,3), Cum/1000.
             If (Mod(K,5) .eq. 0) Write (55,160)
         End Do
```

Step 9: **Open flow potentials (*CAOF()*) and total open flow potential (*AOF*) at each time step are printed.**

Note: Prior to printing this information, number of wells in each pay grade for each development type, total number of wells, and two description lines from sub-program TOPOUT() are printed.

```
         Write (55,999)
         Call TOPOUT (ICase, Desc1$, Desc2$)
         Write (55,115)
         Write (55,140) ((Wells(I,J),J=1,3),I=1,3), SumWel
         Write (55,151)
         Do K = 1, MaxTim
```

```
                AOF = 0.
                Do I=1,3
                        Do J=1,3
                                AOF = AOF + CAOF(I,J,K) * Wells(I,J)
                        End Do
                End Do
                Write(55,201) Time(K), ((CAOF(I,J,K),J=1,3),I=1,3),AOF
                If (Mod(K,5) .eq. 0) Write (55,160)
        End Do
```

Step 10: **Bottomhole pressure (*Prbh()*) and wellhead pressure (*Prwh()*) at each time step are printed.**

Note: Prior to printing bottomhole and wellhead pressures, sub-program TOPOUT() is invoked to print description lines.

```
        Write(55,999)
        Call TOPOUT (ICase, Desc1$, Desc2$)
        Write(55,120)
        Write(55,150)
        Do K = 1, MaxTim
                Write(55,200) Time(K), ((Prbh(I,J,K), J=1,3), I=1,3)
                If (Mod(K,5) .eq. 0) Write(55,160)
        End Do
        Write(55,999)
        Call TOPOUT (ICase, Desc1$, Desc2$)
        Write(55,130)
        Write(55,150)
        Do K = 1, MaxTim
                Write(55,200) Time(K), ((Prwh(I,J,K), J=1,3), I=1,3)
                If (Mod(K,5) .eq. 0) Write(55,160)
        End Do
```

Step 11: **Water influx (*WtrInf()*), total water production rate (*Qwtr()*), and cumulative water production (*Wp()*) at each time step are printed.**

Note: Prior to printing this information, sub-program TOPOUT() is invoked to print description lines.

```
        Write(55,999)
        Call TOPOUT (ICase, Desc1$, Desc2$)
        Write(55,135)
        Wp(1) = 0.
        Wp(2) = 0.
        Wp(3) = 0.
        Write(55,153)
        Do K = 1, MaxTim
                DT = Time(K)
                If (K.gt.1) DT = Time(K) - Time(K-1)
                Do I = 1, 3
                        Qwtr(I) = Qw(I,1,K)+Qw(I,2,K)+2.*Qw(I,3,K)
                        Wp(I) = Wp(I) + Qwtr(I) * DT * 365./ 1000.
                        If (Qg(I,1,K).lt.1.) Wp(I) = 0.
                End Do
                Write(55,200) Time(K), (WtrInf(J,K),Wp(J),Qwtr(J),J=1,3)
                If (Mod(K,5) .eq. 0) Write(55,160)
```

```
         End Do
```

Step 12: **Printout formats are defined.**

```
100     Format ('ßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßß',
     +            'ßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßß')
101     Format (//,
     +  '         Gross Gas     Original      Recovery',/
     +  '  Pay     Recovery   Gas-in-Place    Efficiency',/
     +  ' Grade     (MMcf)       (MMcf)        (% OGIP1)',/
     +  ' ÄÄÄÄÄ   ÄÄÄÄÄÄÄÄÄÄ   ÄÄÄÄÄÄÄÄÄÄÄÄ    ÄÄÄÄÄÄÄÄÄÄ')
102     Format (3x, I1, 3x, F10.0, 3x, F10.0, 3x, F10.1,'%')
103     Format (' ÄÄÄÄÄ   ÄÄÄÄÄÄÄÄÄÄ   ÄÄÄÄÄÄÄÄÄÄÄÄ    ÄÄÄÄÄÄÄÄÄÄ',/
     +        1x,'TOTAL', 1x, F10.0, 3x, F10.0, 3x, F10.1,'%',///)
104     Format (' The Field Could No Longer Meet the Rate Constraint',/
     +          ' Beginning in Year ', F5.0)
105     Format (' Automatic Refracturing of Primary Wells Occurred',/,
     +          ' in Year ', F4.0)
106     Format (' Automatic Infilling Was Done for this Case',
     +          ' in Year ', F4.0, /)
107     Format (' Automatic Infilling Was Done for this Case',
     +          ' in Year ', F4.0, /)
110     Format (///,t25,'PRODUCING RATE, MCFD/WELL')
112     Format (///,t22,'CUMULATIVE PRODUCTION, MMCF/WELL')
115     Format (///,t22,'OPEN FLOW POTENTIAL, MCFD/WELL')
120     Format (///,t25,'BOTTOMHOLE PRESSURE, PSIA')
130     Format (///,t26,'WELLHEAD PRESSURE, PSIA')
135     Format (///,t15,'WATER PRODUCTION/INFLUX, MBBL/WELL',
     +          ' AND BPD/WELL')
140     Format (/,' # Wells',10(F7.2,1x))
150     Format (//'               Pay Grade 1            ',
     +          'Pay Grade 2            Pay Grade 3',/,
     +          '    Time ---------------------',
     +          '  ---------------------  ---------------------',/,
     +          '   years Well 1  Well 2  Well 3  Well 1  Well 2',
     +          '  Well 3  Well 1  Well 2  Well 3',/,
     +          '  ------ ------  ------  ------  ------  ------',
     +          '  ------  ------  ------  ------')
151     Format (/'               Pay Grade 1            ',
     +          'Pay Grade 2            Pay Grade 3',/,
     +          '    Time ---------------------',
     +          '  ---------------------  ---------------------',
     +          '  Total',/,
     +          '   years Well 1  Well 2  Well 3  Well 1  Well 2',
     +          '  Well 3  Well 1  Well 2  Well 3   Mcfd',/,
     +          '  ------ ------  ------  ------  ------  ------',
     +          '  ------  ------  ------  ------  ------')
152     Format (/'               Pay Grade 1            ',
     +          'Pay Grade 2            Pay Grade 3',/,
     +          '    Time ---------------------',
     +          '  ---------------------  ---------------------',
     +          '  Total',/,
     +          '   years Well 1  Well 2  Well 3  Well 1  Well 2',
     +          '  Well 3  Well 1  Well 2  Well 3   MMcf',/,
     +          '  ------ ------  ------  ------  ------  ------',
     +          '  ------  ------  ------  ------  ------')
153     Format (//'               Pay Grade 1            ',
     +          'Pay Grade 2            Pay Grade 3',/,
     +          '    Time ---------------------',
     +          '  ---------------------  ---------------------',/,
     +          '   years Influx   Prod    Rate   Influx   Prod ',
     +          '   Rate   Influx   Prod    Rate ',/,
     +          '  ------ ------  ------  ------  ------  ------',
     +          '  ------  ------  ------  ------')
160     Format (1x)
200     Format (1x,F7.3, 9(1x,F7.0))
201     Format (1x,F7.3,10(1x,F7.0))
```

```
999     Format ('\')
```

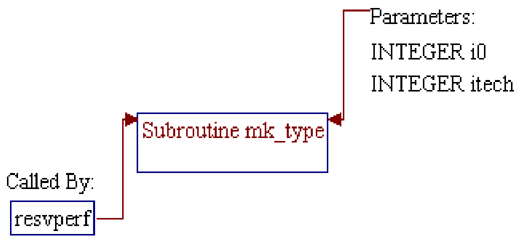
Step 13: **The program control is returned back to the calling routine (sub-program MODULE6()) and the sub-program DATOUT() is ended.**

```
        Return
        End
```

SUB-PROGRAM MK_TYPE()

LOCATION:	MK_TYPE.FOR
MAIN THEME:	This routine creates an input file for the type curve module (MODULE6()).
CALLS:	None
CALLED BY:	RESVPERF (in file RESVPERF.FOR) Main program of Reservoir Performance Module.
READS:	None
CREATES:	[GSAMID].TCI (Type curve input file)

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included. Additional common block and local variables are declared.**

Note: Name of the sub-program is MK_TYPE() and the parameters passed to this sub-program are as follows:

- *i0* Unit number for input file .TCI (unit 56)
- *itech* Technology flag: 1=current, 2=advanced

```
      subroutine mk_type(i0,itech)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'cost.h'
      include 'tech.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
      include 'gsamvar.h'
```

Note: Additional common block and local variables are declared.

```
      integer i,i0,itech,j
      character*80 lines(qline)
      common/ddd/lines
```

Step 2: **Lines 3 and 4 to be printed to file .TCI are set.**

Note: Entries of array variable *lines()* are previously set in sub-program RE_TEMP(). This sub-program reads a user specified template file TEMPLATE.DAT and stores the data in variable array *lines()*. The following code changes entries of *lines(3)* with information such as 11-digit GSAM ID and name of technology (*technm()*). *lines(4)* is set to blank.

```
      lines(3)='GSAM Code: '//gsamid//' Technology: '//technm(itech)
      lines(4)=' '
```

Step 3: **Lines 1 through 9 are printed.**

```
       do i=1,9
        write(i0,555) lines(i)
       enddo
```

Step 4: **Information related to impurities concentrations, gas gravity, temperature, tubing diameter, and flag for speedup are printed.**

Note: Variable names (or value) and descriptions of these parameters are as follows:

• *gasgrv1*	Gas gravity
• *tem*	Temperature (degree F)
• *cnch2s*	Concentration of hydrogen sulfide (fraction)
• *cncco2*	Concentration of carbon dioxide (fraction)
• *cncn2*	Concentration of nitrogen (fraction)
• *cnch2s*	Concentration of hydrogen sulfide (fraction)
• *diam*	Tubing inside diameter (inches)
• *1*	Value for speedup flag. Value of 1 is defaulted in which indicates a speedup run.

```
       write(i0,250)gasgrv1,tem,cnch2s,cncco2,cncn2,diam,1
  250  format(t4,f6.4,t15,f5.0,t24,
      &   f7.2,t34,f7.2,t44,f7.2,t55,f6.3,t68,i1)
```

Step 5: **Lines 11 through 17 are printed.**

```
       do i=11,17
        write(i0,555) lines(i)
       enddo
```

Step 6: **Basic reservoir parameters/properties are printed.**

Note: Variable names and descriptions of these parameters are as follows:

• *i*	Pay grade number
• *pinit()*	Initial reservoir pressure (psia)
• *perm()*	Horizontal permeability (md)
• *permv()*	Vertical permeability (md)

• *poros()*	Total porosity (%)
• *swi()*	Initial water saturation (%)
• *thick()*	Net pay thickness (feet)
• *salin()*	Water salinity (ppm)

```
      do i=1,3
        write(i0,600) i,pinit(i),perm(i),permv(i),
     &  poros(i),swi(i),thick(i),salin(i)
      enddo
 600  format(t4,i1,t10,f6.0,t17,f7.2,t25,
     &   f7.2,t33,f7.2,t44,f5.2,t55,f6.2,t65,f7.0)
```

Step 7: **Lines 21 through 27 are printed.**

```
      do i=21,27
       write(i0,555) lines(i)
      enddo
```

Step 8: **Fractured reservoir properties are printed.**

Note: Variable names and descriptions of these properties are as follows:

• *i*	Pay grade number
• *permm()*	Matrix permeability (md)
• *porma()*	Matrix porosity (%)
• *frcspc()*	Natural fracture spacing (feet)

```
      do i=1,3
        write(i0,800) i,permma(i),porma(i),frcspc(i)
      enddo
 800  format(t4,i1,t10,f7.2,t21,f6.4,t32,f5.2)
```

Step 9: **Lines 31 through 38 are printed.**

```
      do i=31,38
       write(i0,555) lines(i)
      enddo
```

Step 10: **Field development information/parameters are printed.**

Note: Variable names and descriptions of these parameters are as follows:

• *i*	Pay grade number
• *depth1()*	Depth (feet)

• *area()*	Drainage area (acres)
• *wspace()*	Primary well spacing (acres)
• *imod()*	Reservoir Module flags for primary, first infill, and second infill wells
• *rw()*	Wellbore radii of primary, first infill, and second infill wells

```
      do i=1,3
       write(i0,900)
     &   i,depth1(i),area(i),wspace(i),(imod(i,j),j=1,3),
     &   (rw(i,j),j=1,3)
      enddo
 900  format(t4,i1,t8,f6.0,t15,f8.0,t23,f5.0,
     &   t34,i1,t41,i1,t48,i1,t55,f4.2,t61,f4.2,t68,f4.2)
```

Step 11: **Lines 42 through 49 are printed.**

```
      do i=42,49
       write(i0,555) lines(i)
      enddo
```

Step 12: **Fractured and horizontal well data are printed.**

Note: Variable names (or value) and descriptions of these parameters are as follows:

• *i*	Pay grade number
• *0,0,0*	Well types for primary, first infill, and second infill: 0=vertical (default), 1=horizontal
• *halfln()*	Fracture half length or horizontal well length (feet)
• *cond()*	Fracture conductivities of primary, first infill, and second infill wells (md-ft)

```
      do i=1,3
       write(i0,950) i,0,0,0,(halfln(i,j),j=1,3),(cond(i,j),j=1,3)
      enddo
 950  format(t4,i1,t10,i1,t17,i1,t24,i1,t31,f5.0,t38,f5.0,t45,f5.0,
     &   t54,f6.0,t61,f6.0,t68,f6.0)
```

Step 13: **Lines 53 through 61 are printed.**

```
      do i=53,61
       write(i0,555) lines(i)
      enddo
```

Step 14: **Water drive and unconventional reservoir data are printed.**

Note: Variable names and descriptions of these parameters are as follows:

• *i*	Pay grade number
• *kaqtyp()*	Aquifer type based on external reservoir radius to wellbore radius ratio (*Re/Rw*): 0=2.5, 1=5, 2=infinity
• *sgtrap()*	Trapped gas saturation behind advancing water influx front (%)
• *qwmax()*	Maximum water rate. Units and definition depend on the sign and magnitude of *qwmax()*: >1.0 (BPD), 0.0 to 1.0 (fraction of total influx), <0.0 (-BBL/MCF)
• *kuncon()*	Type of unconventional reservoir: 0=dry coal, 1=wet coal, 2=dry shale, 3=wet shale
• *iloc()*	Flag for coal/shale location: 0=Appalacia, 1=Alabama, 2=Western U.S.
• *gascon1()*	Coal/shale gas content (SCF/ton)
• *pl()*	Langmuir pressure (psia)
• *tdes()*	Coal/shale sorption time constant (days)
• *rhoma()*	Matrix (reservoir rock) density (gr/cc)

```
      do i=1,3
       write(i0,690) i,kaqtyp(i),sgtrap(i),qwmax(i),
     &   kuncon(i),iloc(i),gascon1(i),pl(i),tdes(i),rhoma(i)
      enddo
 690  format(t4,i1,t07,i2,5x,f4.2,4x,f5.1,6x,i1,10x,i1,4x,f4.0,6x,
     &       f5.0,5x,f4.0,5x,f4.2)
```

Step 15: **Lines 65 through 72 are printed.**

```
      do i=65,72
       write(i0,555) lines(i)
      enddo
```

Step 16: **Well control parameters are printed.**

Note: Variable names (or value) and descriptions of these parameters are as follows:

• *premin*	User specified minimum wellhead pressure (psia)

• *ratmax*	Maximum gas rate. Units and definition depend on magnitude of *ratmax*: >1.0 (MCFD), <=1.0 (fraction of absolute open flow)
• *timchg*	Starting year to drill infill wells for water drive reservoir (years)
• *1*	First pay grade for the following skin factors
• *skin(1,1,1)*	Skin factor for primary well in pay grade 1
• *skin(1,2,1)*	Skin factor for first infill well in pay grade 1
• *skin(1,3,1)*	Skin factor for second infill well in pay grade 1 (not currently implemented)
• *skin(1,1,2)*	Skin factor for primary well with hydraulic fracture (auto refrac) in pay grade 1
• *2*	Second pay grade for the following skin factors
• *skin(2,1,1)*	Skin factor for primary well in pay grade 2
• *skin(2,2,1)*	Skin factor for first infill well in pay grade 2
• *skin(2,3,1)*	Skin factor for second infill well in pay grade 2 (not currently implemented)
• *skin(2,1,2)*	Skin factor for primary well with hydraulic fracture (auto refrac) in pay grade 2
• *3*	Third pay grade for the following skin factors
• *skin(3,1,1)*	Skin factor for primary well in pay grade 3
• *skin(3,2,1)*	Skin factor for first infill well in pay grade 3
• *skin(3,3,1)*	Skin factor for second infill well in pay grade 3 (not currently implemented)
• *skin(3,1,2)*	Skin factor for primary well with hydraulic fracture (auto refrac) in pay grade 3

```
       write(i0,960) premin,ratmax,timchg,1,
     &  skin(1,1,1),skin(1,2,1),skin(1,3,1),skin(1,1,2)
 960  format(t3,f6.0,t12,f9.2,t25,f4.1,t36,i1,t42,
     &  f5.1,t48,f5.1,t55,f5.1,t65,f6.1)
       write(i0,970) 2,
     &  skin(2,1,1),skin(2,2,1),skin(2,3,1),skin(2,1,2)
       write(i0,970) 3,
     &  skin(3,1,1),skin(3,2,1),skin(3,3,1),skin(3,1,2)
 970  format(t36,i1,t42,f5.1,t48,f5.1,t55,f5.1,t65,f6.1)
```

Step 17: **Lines 72 through 84 are printed.**

```
      do i=76,85
        write(i0,555) lines(i)
      enddo
 555  format(a)
```

Step 17: **Lines 72 through 84 are printed.**

```
      do i=76,85
       write(i0,555) lines(i)
      enddo
```

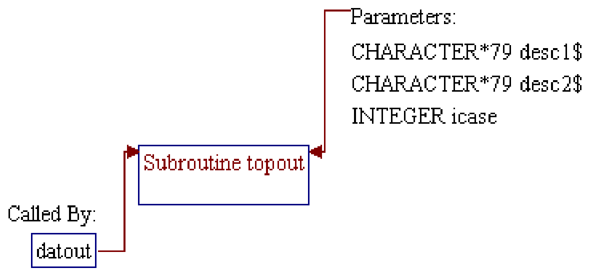
Step 18: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program MK_TYPE() is ended.**

```
      return
      end
```

SUB-PROGRAM TOPOUT()

LOCATION:	MODULE6C.FOR
MAIN THEME:	This routine prints out two header/description lines to type curve output file (.TCO).
CALLS:	None
CALLED BY:	DATOUT() (in file MODULE6D.FOR) Prints out results to the type curve output file.
READS:	None
CREATES:	[GSAM].TCO (Type curve output file)

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Local variables are declared.**

Note: Name of the sub-program is TOPOUT() and the parameters passed to this sub-program are as follows:

• *ICase*	Case number: 1=primary, 2=automatic refrac, 3=automatic infill one time, 4=automatic infill twice (not yet implemented).
• *Desc1$*	First line of description
• *Desc2$*	Second line of description

```
      SUBROUTINE TopOut (ICase,  Desc1$, Desc2$)
```

Note: Local variables are declared.

```
      Character*79 Desc1$, Desc2$
```

Step 2: **Description lines are printed to output file.**

```
      Write (55,100)
      Write (55,101) Desc1$
      Write (55,101) Desc2$
      If (ICase .eq. 1) Write (55,102)
      If (ICase .eq. 2) Write (55,103)
      If (ICase .eq. 3) Write (55,104)
      Write (55,100)
100   Format ('ßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßß',
     +        'ßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßßß')
101   Format (1x,A79)
102   Format (//,1X,'PRIMARY WELLS ONLY, NO INFILLS',/)
103   Format (//,1X,'PRIMARY WELLS ONLY, NO INFILLS,',
     +      ' REFRACED WHEN NEEDED ',/)
104   Format (//,1X,'ONE AUTOMATIC INFILL EPISODE',/)
```

Step 3: **The program control is returned back to the calling routine (sub-program DATOUT()) and the sub-program TOPOUT() is ended.**

```
      Return
      End
```

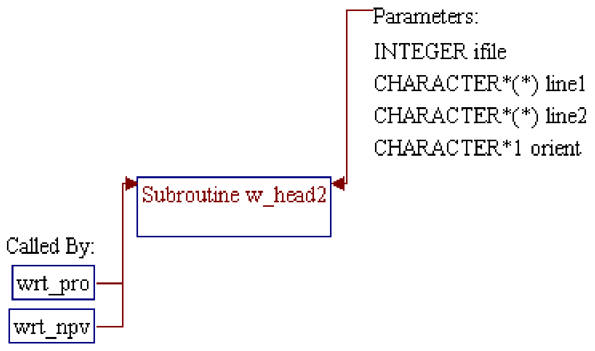
SUB-PROGRAM W_HEAD2()

LOCATION:	IOFUNCT.FOR
MAIN THEME:	This routine prints out two header lines of a table to a specified output file.
CALLS:	None

CALLED BY:	WRT_PRO() (in file WRT_PRO.FOR)
	Writes out cash flow pro-forma to output file .PRO.
	
	WRT_NPV() (in file WRT_PRO.FOR)
	Writes out NPV's to output file .NPV.

READS:	None
CREATES:	Variable output file unit number

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variables are declared.**

Note: Name of the sub-program is W_HEAD2() and the parameters passed to this sub-program are as follows:

• *ifile*	Output file unit number of the table
• *line1*	The first header line
• *line2*	The second header line
• *orient*	Orientation flag (‘P’=portrait, ‘L’=landscape)

```
      subroutine w_head2(ifile,line1,line2,orient)
```

Note: Local variables are declared.

```
      character*(*) line1,line2
      character*1 orient
      integer ifile
```

Step 2: **Header lines are printed.**

```
      if(orient.eq.'L') then
       write(ifile,*) ' E &l1o5.45C &k2S'
      else
       write(ifile,*) ' E &l5.45C (0U (sp16.66h7vsb8T'
      endif
      write(ifile,*) ' &d@ (119X'
      if(line1.ne.' ') write(ifile,1200) line1
      if(line2.ne.' ') write(ifile,1200) line2
      write(ifile,1230)
 1200 format(t30,a)
 1220 format(t30,a,' (3@ &k2S',/)
 1230 format(' (3@ &k2S',/)
```

Step 3: **The program control is returned back to the calling routine (sub-program WRT_PRO() or WRT_NPV()) and the sub-program W_HEAD2() is ended.**

```
      return
      end
```

SUB-PROGRAM WRITEBIN()

LOCATION: READONE.FOR

MAIN THEME: This routine writes out type curve outputs to output file .BIN.

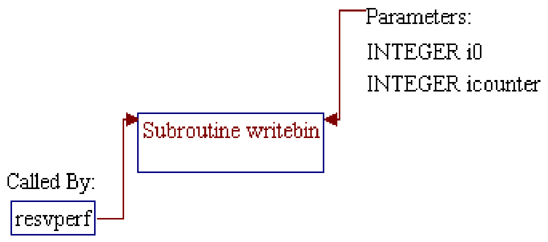
CALLS: None

CALLED BY: RESVPERF (in file RESVPERF.FOR)
Main program of Reservoir Performance Module.

READS: None

CREATES: [GSAM].BIN
(Binary file of type curve results)

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included. Additional common blocks and local variables are declared.**

Note: Name of the sub-program is WRITEBIN() and the parameters passed to this sub-program are as follows:

• *i0*	Unit number of output file .BIN (unit 11)
• *icounter*	Correction year (Undiscovered=0, Discovered=1) (year)

```
      subroutine writebin(i0,icounter)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'gsamvar.h'
      include 'type_out.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
```

Note: Additional common blocks and local variables are declared.

```
      real*4 tgasb(qcase,qpay)
      common/prod_life/ iattt(qcase,qpay)
      Common /stchg1/tgasb
      integer i0,iyr,icase,ipay
```

Step 2: **Working variable for type curve methane gas productions (*tgasb()*) is initialized.**

```
        Do ipay = 1, 3
         Do icase = 1, 3
          tgasb(icase,ipay) = 0.0
         Enddo
        Enddo
```

Step 3: **Methane gas productions (*tgasb()*) are calculated.**

Note: *type_gas(p1,p2,p3)* is calculated total gas production in BCF (including impurities) where *p1*=development type number, *p2*=pay grade number, *p3*=year. Impurities (hydrogen sulfide, carbon dioxide, and nitrogen) are subtracted from the total gas to get the methane volume.

```
         Do ipay = 1, 3
          Do icase = 1, 3
           tgasb(icase,ipay)=tgasb(icase,ipay)+
      @    type_gas(icase,ipay,1)*time(1)*(1.0-h2s-co2-n2)
           do iyr=2,iattt(icase,ipay)
            if ((iyr+icounter).le.qyr) then
            tgasb(icase,ipay)=tgasb(icase,ipay)+
      @     type_gas(icase,ipay,iyr+icounter)*(1.0-h2s-co2-n2)
            endif
           enddo
          Enddo
         Enddo
```

Step 4: **Type curve results are printed to unit file 11 ([GSAM].BIN file).**

Note: Information printed to the .BIN file are:

• *gsamid*	11-digit GSAM code
• *tgasb()*	Methane gas production (BCF)
• *type_ogip()*	Original gas in place (BCF)
• *type_well()*	number of wells
• *kwinyr()*	Number of production years without infills and refrac
• *iattt()*	Production life of the reservoir (years)
• *type_gas()*	Total gas production (BCF)
• *type_pwhp()*	Wellhead pressures of primay wells (psia)

```
           write(i0) gsamid,tgasb,type_ogip,type_well,
      &     kwinyr,iattt,type_gas,type_pwhp
```

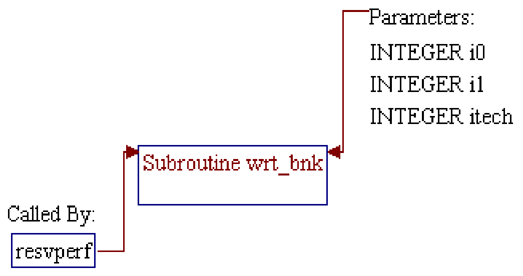
Step 5: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program WRITEBIN() is ended.**

```
       return
       end
```

SUB-PROGRAM WRT_BNK()

LOCATION:	GSAM_B.FOR
MAIN THEME:	This routine reports reserves, OGIP, etc. and summary of economics to output file .DEC. Also reports summary of current technology to output file .SUM or summary of advanced technology to output file .ASM.
CALLS:	None
CALLED BY:	RESVPERF (in file RESVPERF.FOR) Main program of Reservoir Performance Module.
READS:	None
CREATES:	[GSAM].DEC (Summary of economics) [GSAM].SUM (Summary of current technology) [GSAM].ASM (Summary of advanced technology)

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included. Additional common blocks and local variables are declared.**

Note: Name of the sub-program is WRT_BNK() and the parameters passed to this sub-program are as follows:

• *i0*	Unit number for output file .DEC (unit 69)
• *itech*	Technology flag (1=current, 2=advanced)
• *i1*	Unit number for output file .SUM (unit 70) or .ASM (unit 79)

```
      subroutine wrt_bnk(i0,itech,i1)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'global.h'
      include 'cashflow.h'
      include 'costing.h'
      include 'cost.h'
      include 'tech.h'
      include 'field.h'
      include 'tax_nat.h'
      include 'tax_reg.h'
      include 'unitcost.h'
      include 'gsamvar.h'
      include 'welldata.h'
      include 'type_out.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
```

Note: Additional common blocks and local variables are declared.

```
      common/prod_life/ iattt(qcase,qpay)
      common/stchg/iwin_yr
      real*4 temp,tgas
      integer i0,itech,icase,ipay
      character*1 pick(qcase,qpay) !lowest masp
```

Step 2: **Header lines for output file .DEC are printed.**

```
       write(i0,'(20a)')
```

```
     &   '                                             ',
     &   '   #                 Tot.',
     &   '     NPV     NPV      NPV       NPV',
     &   '        NPV         NPV     Chg.',
     &   '    Chg.     Chg.     Drill   Non-Drl       H2O   Window',
     &   '    Productive'
        write(i0,'(20a)')
     &   ' GSAM ID    EIACODE CASE      Resv.   ',
     &   'OGIP   Wells      MASP     Cap.',
     &   '     Prod.   Exp.     Inv.     Drill       Non-Drl',
     &   '     Tax     Exp.',
     &   '    Inv.     Tax      Slope    Slope  Feet  Depth  Year ',
     &   '       Life'
```

Step 3: **Character Array *pick()* is assigned.**

Note: *pick(p1,p2)* is a two dimensional array to indicate which development type in each pay grade has the lowest MASP (Minimum Acceptable Supply Price), where *p1*=developent type and *p2*=pay grade. The value of *pick(p1,p2)* is "*" if the MASP of the development type *p1* in pay grade *p2* is the lowest. Otherwise, a white space is assigned to the *pick(p1,p2)*. This character will be helpful in analyzing the economics.

```
        do icase=1,qcase
         do ipay=1,qpay
           pick(icase,ipay)=' '
         enddo
        enddo
        do ipay=1,3
         temp=masp(1,ipay)
         pick(1,ipay)='*'
         itemp=1
         do icase=2,3
          if(masp(icase,ipay).lt.temp) then
           temp=masp(icase,ipay)
           pick(icase,ipay)='*'
           pick(itemp,ipay)=' '
           itemp=icase
          endif
         enddo
        enddo
```

Step 4: **Technically recoverable methane reserve *tgas* is calculated.**

```
        tgas = 0.0
        do ipay=1,3
         do icase=1,3
         tgas=type_gas(icase,ipay,1)*time(1)*(1.0-h2s-co2-n2)
         if (type_ogip(icase,ipay).gt.99999.0) type_ogip(icase,ipay)=0.
          do iyr=2,iattt(icase,ipay)
           tgas=tgas+type_gas(icase,ipay,iyr)*(1.0-h2s-co2-n2)
          enddo
```

Step 5: **Value of window year (*iwin_yr*) is checked and it set between 2 and *qyr* (*qyr*=140) years if the value is out of range.**

Note: *iwin_yr* is a production year without infills and refrac.

```
          kwinyr(icase,ipay)=iwin_yr
          if(iwin_yr.le.1)kwinyr(icase,ipay)=2
          if(iwin_yr.ge.iattt(icase,ipay))kwinyr(icase,ipay)=qyr
```

Step 6: **Print out summaries of economics, current technology, and advanced technology by pay grade for the primary well case (*icase=1*).**

Note: Information printed to the .SUM or .ASM files are:

• *gsamid*	11-digit GSAM code
• *technm()(1:1)*	First character of technology name (C=current, A=advanced)
• *ipay*	Pay grade number
• *tgas*	Technically recoverable reserves (BCF)
• *type_ogip()*	Original gas in place (BCF)
• *type_well()*	Number of wells could be drilled
• *masp()*	Minimum acceptable supply price ($/MCF)
• *npv_drl()*	NPV of drilling costs, no exploration costs ($MM)
• *npv_tax()*	NPV of total taxes paid (federal, state, severance) ($MM)
• *npv_tax(2,...,...)- npv_tax(1,...,...)*	Difference in NPV of taxes when calculated at $5/MCF and $2/MCF ($MM)

Note: Information printed to the .DEC file are:

• *gsamid*	11-digit GSAM code
• *eiacod*	8-digit EIA code
• *technm()(1:1)*	First character of technology name (C=current, A=advanced)
• *ipay*	Pay grade number
• *casename()(1:1)*	First character of development type name (P=primary, R=refrac, I=infill)
• *pick()*	One character to indicate which development type has the lowest MASP
• *tgas*	Technically recoverable reserves (BCF)
• *type_ogip()*	Original gas in place (BCF)

- *type_well()* Number of wells could be drilled
- *masp()* Minimum acceptable supply price ($/MCF)
- *tot_cap_2()* Total capital at gas price of $2/MCF ($MM)
- *npv_prd()* NPV of gas production ($MM)
- *npv_exp()* NPV of total expenses ($MM)
- *npv_inv()* NPV of total investments ($MM)
- *npv_drl()* NPV of drilling costs, no exploration costs ($MM)
- *npv_inv(1,...,...)- npv_drl(1,...,...)* Difference in NPV between total investment and total tax at gas price of $2/MCF ($MM)
- *npv_tax()* NPV of total taxes paid (federal, state, severance) ($MM)
- *npv_exp(2,...,...)- npv_exp(1,...,...)* Difference in NPV of total expenses when calculated at $5/MCF and $2/MCF ($MM)
- *npv_inv(2,...,...)- npv_inv(1,...,...)* Difference in NPV of total investments when calculated at $5/MCF and $2/MCF ($MM)
- *npv_tax(2,...,...)- npv_tax(1,...,...)* Difference in NPV of taxes when calculated at $5/MCF and $2/MCF ($MM)\
- *slope1()* Slope of NPV due to change only in drilling cost (drilling slope)
- *slope2()* Slope of NPV due to changes in all non-drilling cost (non-drilling slope)
- *depth* Well depth (feet)
- *h2odep* Depth of water in the reservoir (feet)
- *kwinyr()* Production year under primary wells (no infill no refract) (years)
- *iattt()* Production life (years)

```
           if(icase.eq.1)
     &     write(i1,'(a,1x,a1,4x,i1,f9.3,3x,f9.3,3x,f7.0,3x,
     &      f6.2,1x,3(f9.3,1x)))')
     &     gsamid,technm(itech)(1:1),ipay,tgas,type_ogip(icase,ipay),
     &     type_well(icase,ipay),masp(icase,ipay),npv_drl(1,icase,ipay),
     &      npv_tax(1,icase,ipay),
     &     (npv_tax(2,icase,ipay)-npv_tax(1,icase,ipay))
          write(i0,'(a,1x,a,2x,a,1x,i1,1x,a,1x,a1,1x,f7.1,1x,f7.1,1x,
     &        f6.0,1x,f6.2,1x,f8.1,6(f9.3,1x),5(f8.3,1x),f6.0,1x,f6.0,
     &      1x,i3,8x,i3)')
     &      gsamid,eiacod,technm(itech)(1:1),ipay,casename(icase)(1:1),
     &      pick(icase,ipay),
     &      tgas,type_ogip(icase,ipay),
     &      type_well(icase,ipay),masp(icase,ipay),
     &      tot_cap_2(icase,ipay),npv_prd(icase,ipay),
     &      npv_exp(1,icase,ipay),npv_inv(1,icase,ipay),
     &      npv_drl(1,icase,ipay),npv_inv(1,icase,ipay)-
     &      npv_drl(1,icase,ipay),
     &    npv_tax(1,icase,ipay),
     &    npv_exp(2,icase,ipay)-npv_exp(1,icase,ipay),
     &    npv_inv(2,icase,ipay)-npv_inv(1,icase,ipay),
     &    (npv_tax(2,icase,ipay)-npv_tax(1,icase,ipay)),
     &    slope1(icase,ipay),slope2(icase,ipay),depth,
     &    h2odep,kwinyr(icase,ipay),iattt(icase,ipay)
```

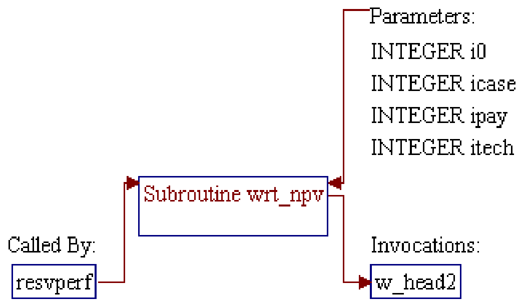
Step 7: **Loops of pay grade and development type are closed. The program control is returned back to the calling routine (program RESVPERF) and the sub-program WRT_BNK() is ended.**

```
        enddo
       enddo
      return
      end
```

SUB-PROGRAM WRT_NPV()

LOCATION:	WRT_PRO.FOR
MAIN THEME:	This routine writes out net present values (NPV's) to output file .NPV.
CALLS:	W_HEAD2() (in file GSAM_B.FOR) Prints out two header lines of a table to a specified output file.
CALLED BY:	RESVPERF (in file RESVPERF.FOR) Main program of Reservoir Performance Module.
READS:	None
CREATES:	[GSAMID].NPV (Net present values)

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included. Local variables are declared.**

Note: Name of the sub-program is WRT_NPV() and the parameters passed to this sub-program are as follows:

• *i0*	Unit number for output file .NPV (unit 33)
• *itech*	Technology flag (1=current, 2=advanced)
• *ICase*	Case number: 1=primary, 2=automatic refrac, 3=automatic infill one time.
• *ipay*	Pay grade number

```
      subroutine wrt_npv(i0,itech,icase,ipay)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'global.h'
      include 'field.h'
      include 'cashflow.h'
      include 'cost.h'
      include 'tech.h'
      include 'costing.h'
      include 'tax_nat.h'
      include 'tax_reg.h'
      include 'npv.h'
      include 'gsamvar.h'
```

Note: Local variables are declared.

```
      integer i0,inpv,itech,icase,ipay
      character*80 line80
      character*2 ch2
```

Step 2: **Value from pay grade code *ipay* is assigned to 2-digit character variable *ch2*.**

```
      write(ch2,'(i2)') ipay
```

Step 3: **String variable *line80* is set.**

Note: *line80* is printed as a header line in output file .NPV. Information written to this variable includes 11-digit GSAM ID, name of technology, and pay grade number.

```
        line80=
      & 'GSAM ID: '//gsamid//' Tech.: '//technm(itech)//
      &  ' Case: '//casename(icase)//' P.G.: '//ch2
```

Step 4: **Header lines are printed.**

Note: Sub-program W_HEAD2() is invoked to print the first two header lines. String variable *line80* is passed to W_HEAD2() and printed as the second line. Character *'P'* passed to W_HEAD2() is an indicator to print these header lines with orientation portrait.

```
         call w_head2(i0,'NPV Calculations',line80,'P')
```

Note: Another header line for titles of four cases of the NPV's are printed.

```
         write(i0,*)
         write(i0,'(t55,a,t75,a,t95,a,t115,a)')
      &     'Regular Case','+$1 Mcf','Zero Drill Cost',
      &     'All Other Costs Zero'
```

Step 5: **Net present values are printed.**

Note: The .NPV file is a five column table where the first column is the name of the NPV component followed by values of the four cases (regular, +$1 MCF, zero drilling cost, and zero all other costs) for that particular NPV. Note that some NPV components only have regular case.

```
         write(i0,*)
         write(i0,1014) 'NPV of cashflow ($MM)', (npv(inpv),inpv=1,4)
         write(i0,*)
         write(i0,1014) 'NPV Gas Prod. less Roy. and Sev. Tax (bcf)',
      &    (g_prd_npv(inpv),inpv=1,4)
         write(i0,1014) 'NPV Oil Prod. less Roy. and Sev. Tax (MMBbl)',
      &    (o_prd_npv(inpv),inpv=1,4)
         write(i0,1014) 'NPV of Gross Sales less Royalties ($MM)',
      &    (gross_npv(inpv),inpv=1,4)
         write(i0,1014) 'NPV of Expenses ($MM)',
      &    (toc_npv(inpv),inpv=1,4)
         write(i0,1014)
      &    'NPV of Tang. Investments (Excluding Drilling) ($MM)',
      &    (tan_npv(inpv),inpv=1,4)
         write(i0,1014)
      &    'NPV of Intang. Investments (Excluding Drilling) ($MM)',
      &    (int_npv(inpv),inpv=1,4)
         write(i0,1014) 'NPV of Development Well Costs ($MM)',
      &    (dwc_npv(inpv),inpv=1,4)
         write(i0,1014) 'NPV of Exploratory Well Costs ($MM)',
      &    (ewc_npv(inpv),inpv=1,4)
         write(i0,1014) 'NPV of State and Federal Taxes ($MM)',
     &    (tax_npv(inpv),inpv=1,4)
          write(i0,1014) 'NPV of Depletable G&G/Lease ($MM)',
     &    (depggla_npv(inpv),inpv=1,4)
          write(i0,1014) 'NPV of Expensed G&G/Lease ($MM)',
     &    (expggla_npv(inpv),inpv=1,4)
          write(i0,1014) 'NPV of Federal Tax Credits ($MM)',
     &    (credit_npv(inpv),inpv=1,4)
          write(i0,*)
          write(i0,1014) 'NPV of Project:',
     &     (gross_npv(inpv)-toc_npv(inpv)-
     &     int_npv(inpv)-tan_npv(inpv)-dwc_npv(inpv)-ewc_npv(inpv)-
     &     tax_npv(inpv)-depggla_npv(inpv)-
     &     expggla_npv(inpv)+credit_npv(inpv),inpv=1,4)
          write(i0,1014) 'Total Cost of Proj:',
     &     (totalcst(inpv),inpv=1,4)
          write(i0,*)
          write(i0,*)
          write(i0,1015) 'Inc. in NPV WRT $1 Inc. in Gas Price: ',
     &    npv(2)-npv(1)
          write(i0,1015)
     &    'Inc. in NPV WRT 1 Million Dollar Drop in Drilling Cost: ',
     &     (npv(3)-npv(1))/(totalcst(4))
          write(i0,1015)
     &    'Inc. in NPV WRT 1 Million Dollar Drop in All Other Cost: ',
     &     (npv(4)-npv(1))/(totalcst(3))
1014  format(t1,a,t55,f12.4,t75,f12.4,t95,f12.4,t115,f12.4)
1015  format(t1,a,t58,f7.3)
```

Step 6: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program WRT_PRO() is ended.**

```
      return
      end
```

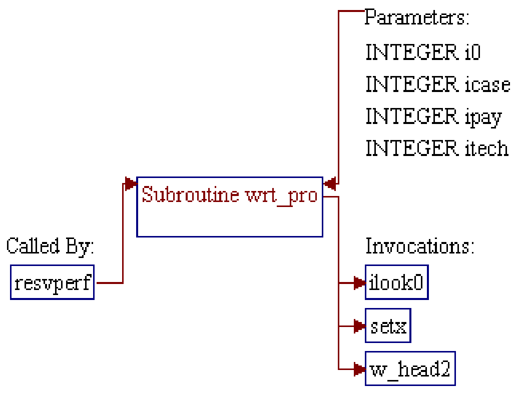
SUB-PROGRAM WRT_PRO()

LOCATION:	WRT_PRO.FOR
MAIN THEME:	This routine writes out cash flow pro-forma to output file .PRO.

CALLS:	ILOOK0() (in file IOFUNCT.FOR)
	Searches location of an integer number in a set of array.
	
	SETX() (in file IOFUNCT.FOR)
	Initializes a real array with a specified value.
	
	W_HEAD2() (in file GSAM_B.FOR)
	Prints out two header lines of a table to a specified output file.

CALLED BY:	RESVPERF (in file RESVPERF.FOR) Main program of Reservoir Performance Module.
READS:	None
CREATES:	[GSAMID].PRO (Cash flow pro-forma)

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included. Local variables are declared.**

Note: Name of the sub-program is WRT_PRO() and the parameters passed to this sub-program are as follows:

• *i0*	Unit number for output file .PRO (unit 31)
• *itech*	Technology flag (1=current, 2=advanced)
• *ICase*	Case number: 1=primary, 2=automatic refrac, 3=automatic infill one time.
• *ipay*	Pay grade number

```
        subroutine wrt_pro(i0,itech,icase,ipay)
```

Note: Header .h files which declare global variables and common blocks are included.

```
        include 'dimen.h'
        include 'global.h'
        include 'field.h'
        include 'cashflow.h'
        include 'costing.h'
        include 'tax_nat.h'
        include 'tax_reg.h'
        include 'cost.h'
        include 'tech.h'
        include 'gsamvar.h'
```

Note: Local variables are declared.

```
        integer iyr1,iyr2,i0,numcol,npage,ipage,iyr,itech,icase,ipay
        character*80 line80
        character*2 ch2
        integer nyr1
        real*4 toc_mcf(qyr)
```

Step 2: **Sub-program ILOOK0() is invoked to search for location of region identifier in array *tax_st()* which corresponds to state code *state.***

```
        call ilook0(state,tax_st,ntax_st,istate)
```

Step 3: **Value from pay grade code *ipay* is assigned to 2-digit character variable *ch2.***

```
        write(ch2,'(i2)') ipay
```

Step 4: **Total operating cost per MCF of gas produced (*toc_mcf()*) is calculated.**

Note: First, sub-program SETX() is invoked to zero out array variable *toc_mcf()*, then the cost is calculated by dividing total operating cost (*toc()*) with total gas production (*gasprod()*).

```
      call setx(toc_mcf,qyr,0)
      do iyr=1,nyr
       if(gasprod(iyr).gt.0) toc_mcf(iyr)=toc(iyr)/gasprod(iyr)
      enddo
```

Step 5: **String variable *line80* is set.**

Note: *line80* is printed as a header line in output file .PRO. Information written to this variable includes 11-digit GSAM ID, name of technology, and pay grade number.

```
      line80=
     & 'GSAM ID: '//gsamid//' Tech.: '//technm(itech)//
     &  ' Case: '//casename(icase)//' P.G.: '//ch2
```

Step 6: **Number of pages to be printed (*npage*) is calculated.**

```
      numcol=7
      nyr1=nyr
      npage=(nyr1)/numcol
      if(mod(nyr1,numcol).gt.0) npage=npage+1
```

Step 7: **Loop for pages is initialized.**

```
      do ipage=1,npage
```

Step 8: **Header lines are printed.**

Note: Sub-program W_HEAD2() is invoked to print the first two header lines. String variable *line80* is passed to W_HEAD2() and printed as the second line. Character '*P*' passed to W_HEAD2() is an indicator to print these header lines with orientation portrait. A word "*Continued*" is added to the first header line if this is not the first page.

```
         if(ipage.eq.1) then
           call w_head2(i0,'Detailed Financial Report',line80,'P')
         else
           call w_head2(i0,'Detailed Financial Report - Continued',
     &          line80,'P')
         endif
```

Note: The beginning and end year numbers (*iyr1* and *iyr2*) for the current page is calculated. The year numbers are then printed to the current page in tabular form.

```
         iyr1=1+numcol*(ipage-1)
         iyr2=min(nyr1,numcol+numcol*(ipage-1))
         write(i0,1011) 'Year',(iyr,iyr=iyr1,iyr2)
         write(i0,2000) ('========',iyr=iyr1,iyr2)
```

Step 9: **Cash flow pro-forma is printed.**

Note: Each page of the output file .PRO is a seven column table where the first column is the component's name of the cash flow followed by six values of that component based on years of the current page. The first page will show values of year 1 to year 6.

```
         write(i0,1013) 'Oil Production (MMBO)',
     &     (oilprod(iyr),iyr=iyr1,iyr2)
         write(i0,1013) 'Gas Production (BCF)',
     &     (gasprod(iyr),iyr=iyr1,iyr2)
         write(i0,101) 'Gross Revenues (MM$)',
     &    (oilprod(iyr)*oprice(iyr)+
     &     gasprod(iyr)*gprice(iyr),iyr=iyr1,iyr2)
         write(i0,102) 'Gravity/Trans. Cost Adj.',
     &    (gravpen(iyr)+transcst(iyr),iyr=iyr1,iyr2)
         write(i0,101) 'Adjusted Revenues',(adjgross(iyr),iyr=iyr1,iyr2)
         write(i0,102) 'Royalties',
     &      (adjgross(iyr)*royrate,iyr=iyr1,iyr2)
         write(i0,101) 'Net Sales',(netsales(iyr),iyr=iyr1,iyr2)
         write(i0,101) 'Total Operating Cost',(toc(iyr),iyr=iyr1,iyr2)
         write(i0,101) 'Operating Cost/Mcf',
     &     (toc_mcf(iyr),iyr=iyr1,iyr2)
         write(i0,102) 'G&A on Expensed Items',
     &     (ga_exp(iyr),iyr=iyr1,iyr2)
         write(i0,102) 'G&A on Capitalized Items',
     &     (ga_cap(iyr),iyr=iyr1,iyr2)
         write(i0,102) 'Pressure Maint./Cycling',
     &     (inj(iyr),iyr=iyr1,iyr2)
         write(i0,102) 'General O&M',(oam(iyr),iyr=iyr1,iyr2)
         write(i0,102) 'Environmental O&M Costs',(eoam(iyr),iyr=iyr1,iyr2)
         write(i0,102) 'Stimulation Costs',(stim(iyr),iyr=iyr1,iyr2)
         write(i0,102) 'Recompletion Costs',(recomp(iyr),iyr=iyr1,iyr2)
         write(i0,101) 'Intangible Investment',
     &     (ii(iyr),iyr=iyr1,iyr2)
         write(i0,102) 'Intang. Exploratory Costs',
     &       (intang_ewc(iyr),iyr=iyr1,iyr2)
         write(i0,102) 'Intang. Development Costs',
     &       (intang_dwc(iyr),iyr=iyr1,iyr2)
         write(i0,102) 'Other Intangible Costs',
     &     (icap(iyr),iyr=iyr1,iyr2)
```

```
      write(i0,102) 'Environmental Intangible Capital Costs',
     &   (eicap(iyr),iyr=iyr1,iyr2)
      write(i0,101) 'Portion of Intangibles to Capitalize',
     &   (intcap(iyr),iyr=iyr1,iyr2)
      write(i0,101) 'TOTAL INVESTMENTS',
     &   (ti(iyr)+ii(iyr),iyr=iyr1,iyr2)
      write(i0,101) 'Tangible Investments',
     &   (ti(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'Tang. Exploratory Cost',
     &   (tang_ewc(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'Tang. Development Cost',
     &   (tang_dwc(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'Environmental',
     &   (etcap(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'Other Tangible Capital',
     &   (otc(iyr),iyr=iyr1,iyr2)
      write(i0,101) 'Depreciable/Capitalized Investments',
     &   (tci(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'Adj. for Federal Tax Credits',
     &   (tciadj(iyr),iyr=iyr1,iyr2)
      write(i0,101) 'Depreciable/Capitalize Base',
     &   (cap_base(iyr),iyr=iyr1,iyr2)
      write(i0,101) 'Depreciation',
     &   (depr(iyr),iyr=iyr1,iyr2)
      write(i0,101) 'Depletable G&G/Lease Costs',
     &   (la(iyr)*plac+gg(iyr)*pggc,iyr=iyr1,iyr2)
      write(i0,102) 'Lease Acq. Cost',
     &   (la(iyr)*plac,iyr=iyr1,iyr2)
      write(i0,102) 'G&G Costs',
     &   (gg(iyr)*pggc,iyr=iyr1,iyr2)
      write(i0,102) 'Adjustments for Federal Tax Credits',
     &   (dep_crd(iyr),iyr=iyr1,iyr2)
      write(i0,101) 'Depletion Base',
     &   (dggla(iyr),iyr=iyr1,iyr2)
      write(i0,101) 'Expensed G&G/Lease Costs',
     &   (eggla(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'Lease Purchase Cost',
     &   (la(iyr)*(1-plac),iyr=iyr1,iyr2)
      write(i0,102) 'G&G Costs',
     &   (gg(iyr)*(1-pggc),iyr=iyr1,iyr2)
      write(i0,101) 'Net Revenues',
     &   (netsales(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'Operator Severance Taxes',
     &   (sevtax(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'Operating Costs',
     &   (toc(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'Expensed Int.,G&G, and Lease Acq.',
     &   (ii(iyr)-intcap(iyr)+eggla(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'Depreciation',
     &   (depr(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'Depletion Allowance',
     &   (deplet(iyr),iyr=iyr1,iyr2)
      write(i0,101) 'Taxable Income',
     &   (nibta(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'Tax Credit Addback',
     &   (eortca(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'Intangible Addback',
     &   (intadd(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'G&G/Lease Addback',
     &   (ggla(iyr),iyr=iyr1,iyr2)
      write(i0,101) 'Net Income Before Taxes',
     &   (nibt(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'State Income Taxes',
     &   (sttax(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'Federal Income Tax',
     &   (fedtax(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'Federal Tax Credits',
     &   (fedtaxc(iyr),iyr=iyr1,iyr2)
      write(i0,101) 'Net Income After Taxes',
     &   (niat(iyr),iyr=iyr1,iyr2)
      write(i0,102) 'plus Depreciation',
     &   (depr(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'plus Depletion',
     &   (deplet(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'less Depletable Items',
     &   (dggla(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'less Depreciable/Capitalized Items',
     &   (intcap(iyr)+ti(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'less Tax Credit on Expensable Items',
     &   (eortca(iyr)+intadd(iyr)+ggla(iyr),iyr=iyr1,iyr2)
       write(i0,101) 'Annual After Tax Cash Flow',
     &   (aatcf(iyr),iyr=iyr1,iyr2)
       write(i0,101) 'Discounted After Tax Cash Flow',
     &   (datcf(iyr),iyr=iyr1,iyr2)
       write(i0,101) 'Cumulative Discounted After Tax Cash Flow',
     &   (catcf(iyr),iyr=iyr1,iyr2)
```

Step 10: **Loop for pages is closed.**

```
       enddo
```

Step 11: **Formats for printing out cash flow are declared.**

```
 1011 format(t40,a,t50,20(6x,i2,2x))
 1013 format(t1,a,t50,20(1x,f8.3,1x))
 101  format(t1,a,t50,20(1x,f8.2,1x))
 102  format(t2,a,t50,20(1x,f8.2,1x))
 103  format(t3,a,t50,20(1x,f8.0,1x))
 104  format(t4,a,t50,20(1x,f8.0,1x))
 105  format(t5,a,t50,20(1x,f8.0,1x))
 2000 format(t50,20(1x,a,1x))
```

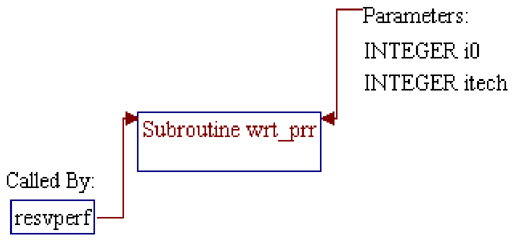
Step 12: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program WRT_PRO() is ended.**

```
       return
       end
```

SUB-PROGRAM WRT_PRR()

LOCATION:	GSAM_A.FOR
MAIN THEME:	This routine writes out a reduced form of cash flow pro-forma to output file .PRR.
CALLS:	None
CALLED BY:	RESVPERF (in file RESVPERF.FOR) Main program of Reservoir Performance Module.
READS:	None
CREATES:	[GSAMID].PRP (Reduced form of cash flow pro-forma)

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included. Local variables are declared.**

Note: Name of the sub-program is WRT_PRR() and the parameters passed to this sub-program are as follows:

• *i0*	Unit number for output file .PRR (unit 67)
• *itech*	Technology flag (1=current, 2=advanced)

```
      subroutine wrt_prr(i0,itech)
```

Note: Header .h files which declare global variables and common blocks are included.

```
      include 'dimen.h'
      include 'global.h'
      include 'cashflow.h'
      include 'costing.h'
      include 'cost.h'
      include 'tech.h'
      include 'field.h'
      include 'tax_nat.h'
      include 'tax_reg.h'
      include 'unitcost.h'
      include 'gsamvar.h'
      include 'welldata.h'
      include 'type_out.h'
```

Note: Local variables are declared.

```
      integer i0,itech,icase,ipay
      real*4 temp
      character*1 pick(qcase,qpay)
```

Step 2: **Character Array *pick()* is assigned.**

Note: *pick(p1,p2)* is a two dimensional array to indicate which development type in each pay grade has the lowest MASP (Minimum Acceptable Supply Price), where *p1*=developent type and *p2*=pay grade. The value of *pick(p1,p2)* is "*" if the MASP of the development type *p1* in pay grade *p2* is the lowest. Otherwise, a white space is assigned to the *pick(p1,p2)*. This character will be helpful in analyzing the economics.

```
       do icase=1,qcase
        do ipay=1,qpay
          pick(icase,ipay)=' '
        enddo
       enddo
```

```
        do ipay=1,3
         temp=masp(1,ipay)
         pick(1,ipay)='*'
         itemp=1
         do icase=2,3
          if(masp(icase,ipay).lt.temp) then
           temp=masp(icase,ipay)
           pick(icase,ipay)='*'
           pick(itemp,ipay)=' '
           itemp=icase
          endif
         enddo
        enddo
```

Step 3: **Loop for pay grades (*ipay*) and loop for development types (*icase*) are initialized.**

```
        do ipay=1,3
         do icase=1,3
```

Step 4: **Technically recoverable methane reserve is calculated and stored in variable *temp*.**

```
          temp=0.0
          do iyr=1,nyr
           temp=temp+type_gas(icase,ipay,iyr)*(1.0-h2s-co2-n2)
          enddo
```

Step 5: **Cash flow pro-forma is printed.**

Note: Parameters printed to the .PRR file are:

• *gsamid*	11-digit GSAM code
• *technm()*	Technology name (Current Technology or Advanced Technology)
• *ipay*	Pay grade number
• *casename()*	Name of development type (Primary, Refrac, or Infill)
• *pick()*	One character to indicate which development type has the lowest MASP
• *temp*	Technically recoverable reserves (BCF)
• *type_ogip()*	Original gas in place (BCF)
• *type_well()*	Number of wells could be drilled
• *masp()*	Minimum acceptable supply price ($/MCF)
• *tot_cap_2()*	Total capital at gas price of $2/MCF ($MM)
• *udatcf_2()*	Undiscounted cash flow at gas price of $2/MCF ($MM)

• *datcf_2()*	Discounted after tax cash flow at gas price of $2/MCF ($MM)
• *udbtcf_2()*	Undiscounted before tax cash flow at gas price of $2/MCF ($MM)
• *dbtcf_2()*	Discounted before tax cash flow at gas price of $2/MCF ($MM)
• *tot_cap_5()*	Total capital at gas price of $5/MCF ($MM)
• *udatcf_5()*	Undiscounted cash flow at gas price of $5/MCF ($MM)
• *datcf_5()*	Discounted after tax cash flow at gas price of $5/MCF ($MM)
• *udbtcf_5()*	Undiscounted before tax cash flow at gas price of $5/MCF ($MM)
• *dbtcf_5()*	Discounted before tax cash flow at gas price of $5/MCF ($MM)

```
          write(i0,'(a,2x,a,1x,i1,1x,a,1x,a1,1x,f7.1,1x,f7.1,1x,
     &       f3.0,1x,f6.3,1x,20(f7.1,1x))')
     &     gsamid,technm(itech),ipay,casename(icase),
     &     pick(icase,ipay),
     &     temp,type_ogip(icase,ipay),
     &     type_well(icase,ipay),masp(icase,ipay),
     &     tot_cap_2(icase,ipay),udatcf_2(icase,ipay),
     &     datcf_2(icase,ipay),udbtcf_2(icase,ipay),
     &     dbtcf_2(icase,ipay),
     &     tot_cap_5(icase,ipay),udatcf_5(icase,ipay),
     &     datcf_5(icase,ipay),udbtcf_5(icase,ipay),
     &     dbtcf_5(icase,ipay)
```

Step 6: **Loop for pay grades (*ipay*) and loop for development types (*icase*) are closed.**

```
         enddo
        enddo
```

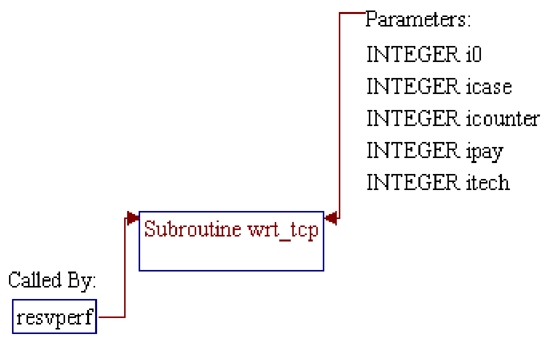
Step 7: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program WRT_PRR() is ended.**

```
         Return
         End
```

SUB-PROGRAM WRT_TCP()

LOCATION:	GSAM_A.FOR
MAIN THEME:	This routine writes out production and operation costs to output file .PRD, file fed to E&P Module.
CALLS:	None
CALLED BY:	RESVPERF (in file RESVPERF.FOR) Main program of Reservoir Performance Module.
READS:	None
CREATES:	[GSAM].PRD (Production and operation costs)

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program are declared. Header ".h" files are included. Additional common block is declared.**

Note: Name of the sub-program is WRT_TCP() and the parameters passed to this sub-program are as follows:

• *i0*	Unit number for output file .PRO (unit 31)
• *itech*	Technology flag (1=current, 2=advanced)
• *ICase*	Case number: 1=primary, 2=automatic refrac, 3=automatic infill one time.
• *ipay*	Pay grade number
• *icounter*	Correction year (Undiscovered=0, Discovered=1) (year)

```
      subroutine wrt_tcp(i0,itech,icase,ipay,icounter)
```

Note: Header .h files which declare global variables and common blocks are included.

```
        include 'dimen.h'
        include 'global.h'
        include 'field.h'
        include 'cost.h'
        include 'tech.h'
        include 'costing.h'
        include 'welldata.h'
        include 'type_out.h'
        include 'gsamvar.h'
```

Note: Additional common block is declared.

```
        common/prod_life/iattt(qcase,qpay)
```

Step 2: **Well production life (*iattt()*) is set to number of years to be analyzed (*nyr*).**

Note: *iattt()* is set to zero if the value is negative (not specified).

```
        iattt(icase,ipay) = nyr
        if (iattt(icase,ipay).le.0) iattt(icase,ipay) = 0
```

Step 3: **Profiles of pressures, operating and maintenance cost, and gas production are printed.**

Note: Profiles of five parameters (in yearly basis) are printed to output file .PRD. These parameters are primary well bottom hole pressure (*'PBHP'*, *type_pbhp()*), primary well wellhead pressure (*'PWHP'*, *type_pwhp()*), infill well bottom hole pressure (*'IBHP'*, *type_ibhp()*), operating and maintenance cost (*'O&M'*), and gas production (*'GASP'*). The following information are printed prior to printing the profiles:

• *gsamid*	11-digit GSAM code
• *eiacod*	8-digit EIA code
• *technm()(1:1)*	First character of technology name (C=current, A=advanced)
• *ipay*	Pay grade number
• *casename()(1:1)*	First character of development type name (P=primary, R=refrac, I=infill)

Note that the pressure profiles are printed if the report is requested in input file REGIONS.DAT (*prt_prs=.TRUE.*).

```
      if(prt_prs) then
       write(i0,300)
     &  gsamid,eiacod,technm(itech)(1:1),ipay,casename(icase)(1:1),
     &  'PBHP',iattt(icase,ipay),
     & (type_pbhp(icase,ipay,iyr+icounter),iyr=1,nyr)
       write(i0,300)
     &  gsamid,eiacod,technm(itech)(1:1),ipay,casename(icase)(1:1),
     &  'PWHP',iattt(icase,ipay),
     &  (type_pwhp(icase,ipay,iyr+icounter),iyr=1,nyr)
       write(i0,300)
     &  gsamid,eiacod,technm(itech)(1:1),ipay,casename(icase)(1:1),
     &  'IBHP',iattt(icase,ipay),
     &  (type_ibhp(icase,ipay,iyr+icounter),iyr=1,nyr)
      endif
       write(i0,301)
     &  gsamid,eiacod,technm(itech)(1:1),ipay,casename(icase)(1:1),
     &  'O&M ',iattt(icase,ipay),
     &  (totoam(iyr),iyr=1,nyr)
      write(i0,302)
     &  gsamid,eiacod,technm(itech)(1:1),ipay,casename(icase)(1:1),
     &  'GASP',iattt(icase,ipay),
     &  (type_gas(icase,ipay,iyr+icounter)*(1.0-co2-n2-h2s),iyr=1,nyr)
 300  format(a,3x,a,1x,a,1x,i1,1x,a,1x,a,1x,i2,1x,60(f8.0,1x))
 301  format(a,3x,a,1x,a,1x,i1,1x,a,1x,a,1x,i2,1x,60(f9.4,1x))
 302  format(a,3x,a,1x,a,1x,i1,1x,a,1x,a,1x,i2,1x,60(f9.4,1x))
```

Step 4: **The program control is returned back to the calling routine (program RESVPERF) and the sub-program WRT_TCP() is ended.**

```
        Return
        End
```

SUB-PROGRAM CHKDIM()

LOCATION: IOFUNCT.FOR

MAIN THEME: This routine checks if dimension of an array has been exceeded. This routine is used to avoid error due to accessing outside the range of fixed size arrays.

CALLS: None

CALLED BY: RD_COST() (in file READINP.FOR)
Reads COST.DAT which contains costs related information.

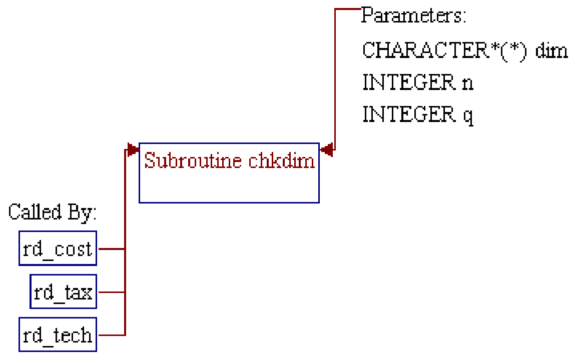
RD_TAX_NAT() (in file READINP.FOR)
Reads TAX_NAT.DAT which contains information about the national level tax assumptions.

RD_TECH() (in file READINP.FOR)
Reads TECH.DAT which contains information on number of technologies and data specifications for each technology.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variables are declared.**

Note: Name of the sub-program is CHKDIM() and the parameters passed to this sub-program are as follows:

• *n*	Location/pointer to an array variable to be accessed
• *q*	Size of array variable
• *dim*	A string variable that stores the name of the array variable

```
      subroutine chkdim(n,q,dim)
```

Note: Local variables are declared.

```
      integer n,q
      character*(*) dim
```

Step 2: **An error message is printed to the console and the program is halted if location/pointer (*n*) to access the array is higher than the size of the array (*q*).**

```
      if(n .gt. q) then
        write(6,*) dim,' exceeded - Program must be Recompiled'
        stop
      endif
```

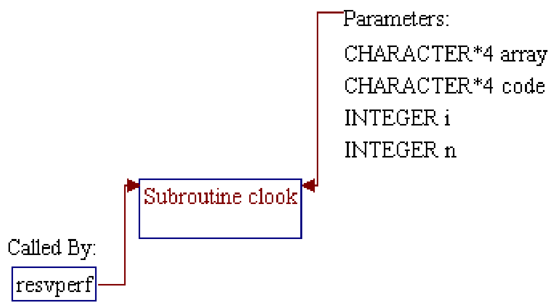
Step 3: **The program control is returned back to the calling routine (sub-program RD_COST(), RD_TAX(), or RD_TECH(),) and the sub-program CHKDIM() is ended.**

```
      return
      end
```

SUB-PROGRAM CLOOK()

LOCATION:	IOFUNCT.FOR
MAIN THEME:	This routine sequentially searches location of a 4-digit code in a set of string array.
CALLS:	None
CALLED BY:	RESVPERF (in file RESVPERF.FOR) Main program of Reservoir Performance Module.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variables are declared.**

Note: Name of the sub-program is CLOOK() and the parameters passed to this sub-program are as follows:

Input Parameters:

•	*code*	4-character string
•	*array()*	Array of strings (each entry is 4-character in size)
•	*n*	Number of data in *array()*

Output Parameter:

•	*i*	Location of *code* in *array()*

```
      subroutine clook(code,array,n,i)
```

Note: Local variables are declared.

```
      integer n,i
      character*4 code,array(*)
```

Step 2: **Location of *code* in *array()* is searched by matching the value of *code* with entries in *array()* in sequential manner. The found location/pointer is stored in variable i. If no match is found, a zero is returned.**

```
      do i=1,n
       if(code.eq.array(i)) return
      enddo
      i=0
```

Step 3: **Program control is returned back to the calling routine (program RESVPERF) and the sub-program CLOOK() is ended.**

```
      return
      end
```

SUB-PROGRAM GETRSP()

LOCATION: IOFUNCT.FOR

MAIN THEME: This routine transforms a YES or NO response (in a form of string variable) to a logical true and false. This routine returns a logical .*TRUE*. if the response string consists of either character “y” or “Y”, or returns .*FALSE*. for “n” or “N”. An error message is printed to a console if the response string does not contain any of the above characters.

CALLS: None

CALLED BY: RD_REGS() (in file GSAM_A.FOR)
Reads REGIONS.DAT file which contains information about the list of the .GSM files to be run through the RP Module and several YES/NO switches as indicators for opening specific files for consistency checks.

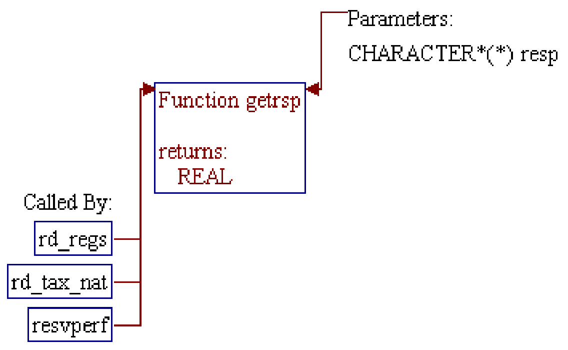
RD_TAX_NAT() (in file READINP.FOR)
Reads TAX_NAT.DAT which contains information about the national level tax assumptions.

RESVPERF (in file RESVPERF.FOR)
Main program of Reservoir Performance Module.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variable are declared.**

Note: Name of the sub-program is GETRSP() and the parameter passed to this sub-program is as follows:

Input Parameter:

- *resp* Response YES/NO string

Output Parameter:

- *getrsp* Logical true (.*TRUE*.) or false (.*FALSE*.)

```
      logical function getrsp(resp)
```

Note: Local variable is declared.

```
      character*(*) resp
```

Step 2: **Variable *getrsp* is first initialized to a logical false (.*FALSE*.).**

```
      getrsp=.false.
```

Step 3: **Logical .*TRUE*. is returned if the response string *resp* consists of either character "Y" or "y". If no character "Y", "y" nor "N", or "n" is found, an error message is printed to the console and the program is halted.**

```
      if(index(resp,'Y').gt.0  .or. index(resp,'y').gt.0) then
        getrsp=.true.
      elseif(index(resp,'N').eq.0 .and. index(resp,'n').eq.0) then
       write(6,*) resp,' not valid answer to YES/NO'
       stop
      endif
```

Step 4: **The program control is returned back to the calling routine (sub-program RD_REGS() or RD_TAX_NAT(), or program RESVPERF) and the sub-program GETRSP() is ended.**

```
      return
      end
```

SUB-PROGRAM ILOOK0()

LOCATION: IOFUNCT.FOR

MAIN THEME: This routine sequentially searches location of an integer number in a set of array.

CALLS: None

CALLED BY: CASHFLOW() (in file CASHFLOW.FOR)
Performs a discounted cash flow analysis (i.e. performs a pro-forma cash flow analysis for every reservoir processed).

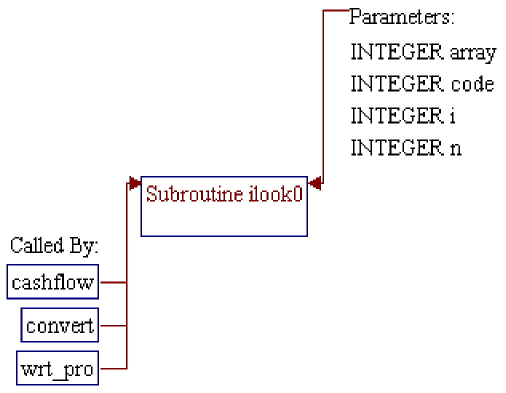
CONVERT() (in file CONVERT.FOR)
Converts the .GSM data into type curve variable names and pay grade level data.

WRT_PRO() (in file WRT_PRO.FOR)
Writes out cash flow pro-forma to output file .PRO.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variables are declared.**

Note: Name of the sub-program is ILOOK0() and the parameters passed to this sub-program are as follows:

Input Parameters:

•	*code*	Integer number to be searched
•	*array()*	Array of integer numbers
•	*n*	Number of data in *array()*

Output Parameter:

•	*i*	Location of *code* in *array()*

```
      subroutine ilook0(code,array,n,i)
```

Note: Local variables are declared.

```
      integer n,i
      integer code,array(*)
```

Step 2: **Location of *code* in *array()* is searched by matching the value of *code* with entries in *array()* in sequential manner. The found location/pointer is stored in variable i. If no match is found, a zero is returned.**

```
      do i=1,n
       if(code.eq.array(i)) return
      enddo
      i=0
```

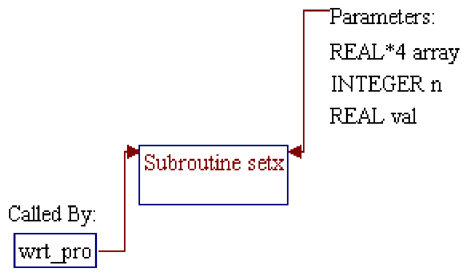
Step 3: **Program control is returned back to the calling routine (sub-program CASHFLOW(), CONVERT(), or WRT_PRO()) and the sub-program ILOOK0() is ended.**

```
      return
      end
```

SUB-PROGRAM SETX()

LOCATION:	IOFUNCT.FOR
MAIN THEME:	This routine initializes a real array with a specified value.
CALLS:	None
CALLED BY:	WRT_PRO() (in file WRT_PRO.FOR) Writes out cash flow pro-forma to output file .PRO.
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variable are declared.**

Note: Name of the sub-program is SETX() and the parameters passed to this sub-program are as follows:

• *array()*	Array of real numbers
• *n*	Number of data in *array()*
• *val*	Real value to be assigned to *array()*

```
      subroutine setx(array,n,val)
```

Note: Local variable is declared.

```
      real*4 array(*)
```

Step 2: **Set all entries in *array()* equal to the value of *val*.**

```
      do 1 i=1,n
        array(i)=val
    1 continue
```

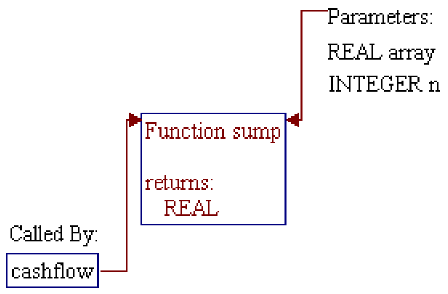
Step 3: **The program control is returned back to the calling routine (sub-program WRT_PRO()) and the sub-program SETX() is ended.**

```
      return
      end
```

SUB-PROGRAM SUMP()

LOCATION:	IOFUNCT.FOR
MAIN THEME:	This routine adds all numbers in a set of a real array.
CALLS:	None
CALLED BY:	CASHFLOW() (in file CASHFLOW.FOR) Performs a discounted cash flow analysis (i.e. performs a pro-forma cash flow analysis for every reservoir processed).
READS:	None
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Name and parameters of the sub-program and local variables are declared.**

Note: Name of the sub-program is SUMP() and the parameters passed to this sub-program are as follows:

Input Parameters:

• *array()*	Array of real numbers
• *n*	Number of data in *array()*

Output Parameter:

• *sump*	Sum of all numbers in *array()*

```
      function  sump(array,n)
```

Note: Local variables are declared.

```
      integer n,i
      real array(n)
```

Step 2: **All entries in *array()* are summed and the total is returned.**

```
      do i=1,n
      sump = sump + array(i)
      enddo
```

Step 3: **Program control is returned back to the calling routine (sub-program CASHFLOW() ()) and the sub-program SUMP() is ended.**

```
      return
      end
```

PROGRAMMER'S GUIDE FOR THE STORAGE RESERVOIR PERFORMANCE MODULE (SRPM) OF THE GAS SYSTEMS ANALYSIS MODEL (GSAM)

FINAL REPORT

Volume IIIb – SRPM Programmer's Guide

For:

U.S. Department of Energy
National Energy Technology Laboratory
Morgantown, West Virginia
Under Contract Number: DE-AC21-92MC28138

By:

ICF Consulting, Inc.
Fairfax, Virginia

February 2001

Table of Contents

Table of Subroutines

STORAGE RESERVOIR PERFORMANCE MODULE (SRPM) PROGRAMMER'S GUIDE

This programmer's guide provides a detailed description of computer code of Storage Reservoir Performance Module (SRPM) of Gas System Analysis Model (GSAM). The guide is divided into sections. Section "Data Dictionary" gives description of global variables used in the SRPM and also gives a cross-reference of each variable that can provide a quick way to visit each use of variables in the SRPM code. Section "I/O Files Dictionary" lists all SRPM input and output files and their descriptions. General logical flow-chart of subroutines of the SRPM is given in section "Flow Chart". The remaining sections of the programmer's guide describe the main program and subroutines of the SRPM with detailed discussion and explanation of each step in the code.

General Structure of the Program Sections

The explanation of each routine in the program sections is started with the name of the routine. If there are parameters passed to the routine, extension "()" is added to the name of the routine. Before the explanation for the code begins, there are four subheadings:

1. **MAIN THEME:**
   Briefly describes the main purpose of the routine.
2. **READS:**
   Lists of input files read by the routine.
3. **CREATES:**
   Lists of output files created by the routine.
4. **ROUTINE INTERACTIONS:**
   Shows the interactions between the calling routines, the routine itself, and the invoked routines in the form of a flow chart. List of parameters passed to the routine (if any) is also given.

These subheadings are followed by detailed explanations for the computer code. Most of the code is explained in steps, i.e., the explanation for a section of related code is delegated in a single step. Between steps, if a certain section of code needs further explanation, "**Note**" is inserted with the relevant explanation.

The code is printed in a box with line numbers (not a FORTRAN line number). One number (starting with number 1 for each routine) is printed for each line of the SRPM code for the purpose of cross-reference of variables given in Section "Data Dictionary".

Program SRPM.EXE

File SRPM.EXE is the executable program of the SRPM. This program is a compilation of one main program ("STORPERF" stored in file STORPERF.FOR), 26 header files

("*.H" files that store global variables and common blocks), and 68 subroutines ("*.FOR" files that store the SRPM code). Names of the header files, the main program, and the subroutines are listed below:

CASHFLOW.H	STORPERF.FOR	INITCOST.FOR	SETX.FOR
COST.H	BW.FOR	INITUNIT.FOR	SOLVER.FOR
COSTING.H	CALCPQ.FOR	MK_TYPE.FOR	TYP_CRV.FOR
DIMEN.H	CALCS.FOR	PD.FOR	UNITCOST.FOR
FIELD.H	CASHFLOW.FOR	PDWFIN.FOR	VISG.FOR
GEOLOGY.H	CHKDIM.FOR	PRECOST.FOR	VISGA.FOR
GLOBAL.H	CLOOK.FOR	PRESUR.FOR	VISGR.FOR
GSAMVAR.H	CLOOK11.FOR	PSI.FOR	VISW.FOR
RD_DATA.H	CLOOK2.FOR	PWELL.FOR	W_HEAD2.FOR
STORLP.H	CNTRL.FOR	RATE1.FOR	WARREN.FOR
TAX_NAT.H	CONVERT.FOR	RATE2.FOR	WRT_DI.FOR
TAX_REG.H	CONVLV.FOR	RD_AFE.FOR	WRT_PRO.FOR
TECH.H	CPOROS.FOR	RD_COST.FOR	WRT_TCP.FOR
TYPE_OUT.H	CRIT.FOR	RD_GEO.FOR	XLNGR4.FOR
TYPE1.H	CWATER.FOR	RD_REGS.FOR	ZEE.FOR
TYPE10.H	DATOUT.FOR	RD_STOR.FOR	ZFAC.FOR
TYPE2.H	ERRFN.FOR	RD_TAX.FOR	ZFACTR.FOR
TYPE3.H	EXPINT.FOR	RD_TECH.FOR	
TYPE4.H	FIND_REG.FOR	RD_TEMP.FOR	
TYPE5.H	FINDSTEP.FOR	RD_WSPAC.FOR	
TYPE6.H	FRICTN.FOR	RDLEVEX.FOR	
TYPE7.H	GET_TYPE.FOR	RDTAXNAT.FOR	
TYPE8.H	GETRSP.FOR	REALGS.FOR	
TYPE9.H	ILOOK0.FOR	RHOW.FOR	
UNITCOST.H	INIT_WEL.FOR	SETUP.FOR	
WELLDATA.H	INITCASH.FOR	SETVAR.FOR	

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
absrns	Absolute roughness of pipe (in)	Declared in	TYPE4.H	1
		Assigned in	TYP_CRV.FOR	23
		Assigned in	SETVAR.FOR	8
		Called by	PWELL.FOR	9
		Called by	TYP_CRV.FOR	25
		Called by	SETVAR.FOR	1
acprod	Well drainage area (acres)	Declared in	GSAMVAR.H	100, 129
		Assigned in	RD_STOR.FOR	96, 97, 106, 118, 126, 140, 166, 169
		Called by	RD_STOR.FOR	107, 109, 119, 124, 131, 144, 152, 168, 170
		Called by	CONVERT.FOR	46
		Called by	STORPERF.FOR	477
acrelim	Approximate acreage reservoir limit (acres)	Declared in	RD_DATA.H	9, 26
		Called by	RD_STOR.FOR	12, 96
acretot	Approximate acreage total (acres)	Declared in	RD_DATA.H	9, 27
		Called by	RD_STOR.FOR	12, 97
adjgross	Adjusted revenues (MM$)	Declared in	CASHFLOW.H	6, 61
		Assigned in	WRT_PRO.FOR	51, 53
		Assigned in	CASHFLOW.FOR	35
		Assigned in	INITCASH.FOR	6
		Called by	CASHFLOW.FOR	37
afe	AFE proportions (fraction)	Declared in	UNITCOST.H	27, 38
		Assigned in	RD_AFE.FOR	10
		Called by	RD_AFE.FOR	9
afename	AFE component name	Declared in	UNITCOST.H	31, 37
		Called by	RD_AFE.FOR	9
apd	Allowable depletion (MM$)	Declared in	CASHFLOW.H	26, 63
		Assigned in	CASHFLOW.FOR	97, 99, 102
		Assigned in	INITCASH.FOR	22
		Called by	CASHFLOW.FOR	104
apigrav	Gas API gravity from storage reservoir database (deg. API)	Declared in	RD_DATA.H	12, 28
		Called by	RD_STOR.FOR	15
area	Well drainage area (acres)	Declared in	TYPE4.H	1
		Assigned in	MK_TYPE.FOR	49
		Assigned in	INIT_WEL.FOR	40
		Assigned in	CONVERT.FOR	46, 53, 56, 57, 139
		Called by	RATE1.FOR	101
		Called by	DATOUT.FOR	79, 86
		Called by	PRECOST.FOR	81, 83
		Called by	CONVERT.FOR	51, 52, 54, 55, 58, 64, 65, 67, 73, 127, 129
		Called by	GET_TYPE.FOR	21
area_fac	Geological factor for pay grade acreage	Declared in	GEOLOGY.H	5, 11

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Called by	CONVERT.FOR	46
		Called by	RD_GEO.FOR	16, 22
avdep	Average well depth (feet)	Declared in	FIELD.H	18, 27
		Assigned in	CONVERT.FOR	126
		Called by	UNITCOST.FOR	27, 30, 55, 56, 79, 84, 91, 95, 98
bhtemp	Bottomhole temperature (deg. F)	Declared in	GSAMVAR.H	73, 126
		Assigned in	RD_STOR.FOR	68
		Called by	RD_STOR.FOR	123
		Called by	CONVERT.FOR	30
caof	Absolute open flow (MCF/D/well)	Declared in	TYPE5.H	3, 7
		Assigned in	SOLVER.FOR	19, 20, 21
		Assigned in	CNTRL.FOR	19
		Called by	DATOUT.FOR	102
capacity	Ultimate storage capacity from storage reservoir database (MMCF)	Declared in	RD_DATA.H	10, 27
		Assigned in	RD_STOR.FOR	125
		Called by	DATOUT.FOR	59
		Called by	RD_STOR.FOR	14, 126, 127, 128, 129, 132, 145
comp_fs	Compressor fuel and gas shrinkage factor (fraction)	Declared in	UNITCOST.H	12, 33
		Assigned in	STORPERF.FOR	559, 560
		Called by	PRECOST.FOR	97
		Called by	STORPERF.FOR	561, 562, 607
comp_vc	Compressor fuel and gas shrinkage factor (fraction)	Declared in	COST.H	20, 51
		Called by	RD_COST.FOR	93, 96
		Called by	STORPERF.FOR	559, 560
comp_w	Compressor cost (MM$/well)	Declared in	UNITCOST.H	11, 33
		Assigned in	UNITCOST.FOR	33, 34, 36, 37
		Called by	PRECOST.FOR	75
cumgas	Cumulative gas production (MCF/well)	Declared in	TYPE5.H	3, 7
		Assigned in	CALCS.FOR	16, 17, 18
		Assigned in	CALCPQ.FOR	31, 34
		Assigned in	CNTRL.FOR	17
		Called by	RATE1.FOR	41
		Called by	CALCPQ.FOR	24, 56, 57
		Called by	DATOUT.FOR	29, 101
		Called by	STORPERF.FOR	273, 308, 348, 401, 440, 504, 537, 556, 607
cumpay	Cumulative production in a pay grade (MMCF)	Declared in	STORLP.H	29, 36
		Assigned in	WRT_TCP.FOR	30
		Assigned in	STORPERF.FOR	550, 555
		Called by	STORPERF.FOR	567, 568, 569, 570, 571, 572
dbwells	Number of wells from storage reservoir database	Declared in	RD_DATA.H	12, 28
		Assigned in	RD_STOR.FOR	23, 107, 119, 168

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Called by	CONVERT.FOR	129
		Called by	RD_STOR.FOR	98, 109, 110, 131, 140, 169
deltat	Time step size (years)	Declared in	TYPE5.H	9, 12
		Assigned in	STORPERF.FOR	56, 62
		Called by	TYP_CRV.FOR	24
		Called by	GET_TYPE.FOR	31
		Called by	CNTRL.FOR	10
deplet	Depletion (MM$)	Declared in	CASHFLOW.H	25, 63
		Assigned in	CASHFLOW.FOR	104
		Assigned in	INITCASH.FOR	21
		Assigned in	WRT_PRO.FOR	131, 153
		Called by	CASHFLOW.FOR	108, 190, 191, 255
depr	Depreciation (MM$)	Declared in	CASHFLOW.H	21, 63
		Assigned in	CASHFLOW.FOR	65
		Assigned in	INITCASH.FOR	17
		Assigned in	WRT_PRO.FOR	103, 129, 151
		Called by	CASHFLOW.FOR	94, 108, 255
depth	Depth of the pay (ft)	Declared in	GSAMVAR.H	58, 123
		Assigned in	RD_STOR.FOR	48, 50, 51, 52, 54
		Called by	RD_STOR.FOR	53, 55, 56, 57, 68
		Called by	CONVERT.FOR	47, 126
depth1	Depth of the pay (ft)	Declared in	TYPE4.H	1
		Assigned in	MK_TYPE.FOR	49
		Assigned in	INIT_WEL.FOR	42
		Assigned in	CONVERT.FOR	47
		Called by	RATE1.FOR	18
		Called by	CALCPQ.FOR	46
dggla	Depletable G&G and lease acquisition (MM$)	Declared in	CASHFLOW.H	22, 63
		Assigned in	CASHFLOW.FOR	74, 76, 80
		Assigned in	INITCASH.FOR	18
		Assigned in	WRT_PRO.FOR	113, 155
		Called by	CASHFLOW.FOR	87, 90, 256
diam	Tubing diameter (ft)	Declared in	TYPE4.H	1
		Assigned in	CONVERT.FOR	35, 36
		Called by	PWELL.FOR	9, 22, 24, 37, 39, 57, 58, 75, 80, 93, 110, 116,
				119, 141, 145
		Called by	MK_TYPE.FOR	25
dpsi	Pseudo-pressure drop (psia^2/cp)	Declared in	TYPE6.H	1
		Assigned in	CONVLV.FOR	39, 40, 41, 74
		Called by	CALCPQ.FOR	41
		Called by	RATE1.FOR	25
dq	Changes in gas rate (MCF/D/well)	Declared in	TYPE5.H	3, 7

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Assigned in	CALCPQ.FOR	30, 33
		Assigned in	CNTRL.FOR	18
		Assigned in	SETVAR.FOR	17
		Called by	CALCPQ.FOR	44
		Called by	CONVLV.FOR	75
eggla	Expensed G&G and lease acquisition cost (MM$)	Declared in	CASHFLOW.H	24, 63
		Assigned in	CASHFLOW.FOR	69
		Assigned in	INITCASH.FOR	20
		Assigned in	WRT_PRO.FOR	115, 127
		Called by	CASHFLOW.FOR	94, 108
eitcr	Environmental intangible tax credit rate (fraction)	Declared in	TAX_NAT.H	41, 51
		Assigned in	RDTAXNAT.FOR	118
		Called by	CASHFLOW.FOR	153, 155, 252
		Called by	RDTAXNAT.FOR	117
env_oam_g	Environmental O&M – gas ($/MCF)	Declared in	UNITCOST.H	20, 34
		Assigned in	INITUNIT.FOR	17
		Assigned in	UNITCOST.FOR	103
		Called by	PRECOST.FOR	94
env_oam_n	Environmental O&M – wells ($/new well /yr)	Declared in	UNITCOST.H	22, 35
		Assigned in	INITUNIT.FOR	20
		Assigned in	UNITCOST.FOR	106
		Called by	PRECOST.FOR	115
env_oam_w	Envrionmental O&M – water ($/BBL)	Declared in	UNITCOST.H	23, 34
		Assigned in	INITUNIT.FOR	18
		Assigned in	UNITCOST.FOR	104
		Called by	PRECOST.FOR	94
envei	Intangible environment well unit cost (MM$/well)	Declared in	UNITCOST.H	6, 35
		Assigned in	INITUNIT.FOR	15
		Assigned in	UNITCOST.FOR	63
		Called by	PRECOST.FOR	118
envet	Tangible environment well unit cost (MM$/well)	Declared in	UNITCOST.H	7, 35
		Assigned in	INITUNIT.FOR	16
		Assigned in	UNITCOST.FOR	64
		Called by	PRECOST.FOR	119
envni	Intangible environment new well unit cost (MM$/well)	Declared in	UNITCOST.H	8, 35
		Assigned in	INITUNIT.FOR	13
		Assigned in	UNITCOST.FOR	61
		Called by	PRECOST.FOR	112
envnt	Tangible environment new well unit cost (MM$/well)	Declared in	UNITCOST.H	9, 35
		Assigned in	INITUNIT.FOR	14
		Assigned in	UNITCOST.FOR	62
		Called by	PRECOST.FOR	113

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
eoam	Environomental O&M cost (MM$)	Declared in	COSTING.H	7, 27
		Assigned in	INITCOST.FOR	9
		Assigned in	PRECOST.FOR	93, 115, 121
		Assigned in	WRT_PRO.FOR	65
		Called by	CASHFLOW.FOR	38, 59, 161, 253
fedrate	Federal income tax rate (fraction)	Declared in	TAX_NAT.H	5, 50
		Assigned in	RDTAXNAT.FOR	10
		Called by	CASHFLOW.FOR	200
		Called by	RDTAXNAT.FOR	9
fedtax	Federal income tax (MM$)	Declared in	CASHFLOW.H	39, 65
		Assigned in	CASHFLOW.FOR	216
		Assigned in	INITCASH.FOR	35
		Assigned in	WRT_PRO.FOR	145
		Called by	CASHFLOW.FOR	218, 220, 254
fedtaxc	Federal tax credits (MM$)	Declared in	CASHFLOW.H	50, 67
		Assigned in	CASHFLOW.FOR	222, 226, 227, 228, 230, 233, 234,
				235, 236, 237, 239, 243, 245, 248,
				250, 252, 253
		Assigned in	INITCASH.FOR	46
		Assigned in	WRT_PRO.FOR	147
		Called by	CASHFLOW.FOR	254
fix	Fixing well cost upfront for existing storage wells (MM$)	Declared in	COSTING.H	18, 29
		Assigned in	PRECOST.FOR	65
		Called by	CASHFLOW.FOR	31
fix_ex	Fixed O&M costs for existing storage ($/MCF)	Declared in	STORLP.H	28, 35
		Called by	RDLEVEX.FOR	8
		Called by	WRT_DI.FOR	22
fix_w	Cost to bring existing well on line (MM$/well)	Declared in	UNITCOST.H	29, 39
		Assigned in	UNITCOST.FOR	58
		Called by	PRECOST.FOR	65
fld	Field name	Declared in	RD_DATA.H	1, 25
		Called by	DATOUT.FOR	51
		Called by	RD_STOR.FOR	10, 29
		Called by	WRT_TCP.FOR	16, 20, 24, 28
fom	Fixed O&M cost for storage ($/MCF)	Declared in	STORLP.H	18, 31
		Assigned in	STORPERF.FOR	612, 629
		Called by	WRT_DI.FOR	37, 42, 45
fom_op1	Fixed O&M cost for storage option 1 ($/MCF)	Declared in	STORLP.H	15, 31
		Assigned in	WRT_DI.FOR	28, 45, 52
		Called by	WRT_DI.FOR	31, 56

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
fom_op2	Fixed O&M cost for storage option 2 ($/MCF)	Declared in	STORLP.H	16, 31
		Assigned in	WRT_DI.FOR	25, 42, 49
		Called by	WRT_DI.FOR	28, 32, 52, 57
fom_op3	Fixed O&M cost for storage option 3 ($/MCF)	Declared in	STORLP.H	17, 31
		Assigned in	WRT_DI.FOR	22, 37
		Called by	WRT_DI.FOR	25, 34, 49, 59
fsttax	Logical flag for forgiveness of state taxes	Declared in	TAX_REG.H	14, 20
		Assigned in	CASHFLOW.FOR	73
		Assigned in	RDTAXNAT.FOR	145
		Called by	CASHFLOW.FOR	173
fti	Federal taxable income (MM$)	Declared in	CASHFLOW.H	38, 65
		Assigned in	CASHFLOW.FOR	178
		Assigned in	INITCASH.FOR	34
		Called by	CASHFLOW.FOR	180, 184, 186
fxoam_w	Fixed O&M (MM$/well)	Declared in	UNITCOST.H	17, 34
		Assigned in	INITUNIT.FOR	10
		Assigned in	UNITCOST.FOR	90, 93
		Called by	PRECOST.FOR	79
ga_exp	G&A on expensed items (MM$/MCF)	Declared in	CASHFLOW.H	9, 61
		Assigned in	CASHFLOW.FOR	38
		Assigned in	INITCASH.FOR	9
		Assigned in	WRT_PRO.FOR	59
		Called by	CASHFLOW.FOR	59
gas_sev	Severance tax rate (fraction)	Declared in	TAX_REG.H	10, 18
		Assigned in	RD_TAX.FOR	17, 24
		Called by	CASHFLOW.FOR	71
gas_sev_p	Severance tax rate ($/MCF)	Declared in	TAX_REG.H	11, 18
		Assigned in	RD_TAX.FOR	18, 25
		Called by	CASHFLOW.FOR	72
gasgrav	Gas specific gravity	Declared in	RD_DATA.H	12, 28
		Called by	RD_STOR.FOR	15, 25
gasgrv	Gas specific gravity	Declared in	GSAMVAR.H	72, 140
		Assigned in	RD_STOR.FOR	25, 64
		Called by	RD_STOR.FOR	123
		Called by	CONVERT.FOR	29
		Called by	ZFAC.FOR	1, 6, 7
gasgrv1	Gas specific gravity	Declared in	TYPE1.H	2
		Assigned in	CONVERT.FOR	29
		Called by	PWELL.FOR	75, 77, 89, 110, 112, 136, 145, 163
		Called by	MK_TYPE.FOR	25
		Called by	CRIT.FOR	1, 3

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Called by	REALGS.FOR	4, 6
		Called by	VISGA.FOR	1, 2
gasinje	Gas injection (BCF/yr)	Declared in	FIELD.H	10, 24
		Assigned in	STORPERF.FOR	565
		Called by	PRECOST.FOR	85
		Called by	CASHFLOW.FOR	61
gasprod	Gas production (BCF/yr)	Declared in	FIELD.H	9, 24
		Assigned in	WRT_PRO.FOR	22, 45, 48
		Assigned in	STORPERF.FOR	564
gassat	Gas saturation (fraction)	Declared in	GSAMVAR.H	70, 126
		Assigned in	RD_STOR.FOR	27, 71, 75, 79, 129, 146, 163
		Called by	RD_STOR.FOR	69, 74, 124, 130, 144, 147,
				151, 152, 164
		Called by	STORPERF.FOR	474
gg	G&G costs (MM$)	Declared in	COSTING.H	12, 28
		Assigned in	INITCOST.FOR	14
		Assigned in	WRT_PRO.FOR	105, 109, 119
		Called by	CASHFLOW.FOR	69, 74, 76, 77, 166, 226, 233, 234
ggla	G&G/lease addback (MM$)	Declared in	CASHFLOW.H	34, 65
		Assigned in	CASHFLOW.FOR	166, 169
		Assigned in	INITCASH.FOR	30
		Assigned in	WRT_PRO.FOR	139, 159
		Called by	CASHFLOW.FOR	172, 257
gid	Storage ID	Declared in	RD_DATA.H	23, 36
		Called by	RD_STOR.FOR	156
		Called by	STORPERF.FOR	114
gprice	Gas selling price, gas fuel usage price ($/MCF)	Declared in	GLOBAL.H	6, 16
		Assigned in	WRT_PRO.FOR	48
		Assigned in	STORPERF.FOR	623, 624
		Called by	UNITCOST.FOR	108, 109, 110
		Called by	CASHFLOW.FOR	36, 61, 71
gravpen	Gravity penalty (MM$)	Declared in	COSTING.H	10, 28
		Assigned in	INITCOST.FOR	12
		Assigned in	WRT_PRO.FOR	50
		Called by	CASHFLOW.FOR	36
gsamid	Storage ID	Declared in	GSAMVAR.H	8, 11
		Assigned in	RD_STOR.FOR	20
		Called by	MK_TYPE.FOR	20
		Called by	RD_STOR.FOR	31, 32, 34, 35, 100, 112, 156
		Called by	STORPERF.FOR	170, 177, 186, 194, 203, 212,
				220,220, 227, 229, 245, 248,
				251, 267, 302, 342, 395, 411,

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
				434, 471, 485, 500, 510
		Called by	UNITCOST.FOR	75
		Called by	WRT_DI.FOR	8, 11, 30, 55
		Called by	WRT_TCP.FOR	16, 20, 24, 28
gsamsr	2-digit storage region	Declared in	GSAMVAR.H	20, 46
		Called by	CONVERT.FOR	75, 93, 97, 121, 124
		Called by	RD_STOR.FOR	31
		Called by	STORPERF.FOR	559
		Called by	UNITCOST.FOR	40, 52
h2osat_fac	Geological factor for water saturation	Declared in	GEOLOGY.H	8, 11
		Called by	CONVERT.FOR	42
		Called by	RD_GEO.FOR	18, 24
h2s	Concentration of H2S (fraction)	Declared in	GSAMVAR.H	77, 140
		Assigned in	RD_STOR.FOR	65
		Assigned in	STORPERF.FOR	232, 236
		Called by	CONVERT.FOR	31
		Called by	STORPERF.FOR	226
halfln	Fracture half length (ft)	Declared in	TYPE8.H	1
		Assigned in	MK_TYPE.FOR	58
		Assigned in	CONVERT.FOR	88, 94, 102
		Called by	CONVLV.FOR	51
		Called by	UNITCOST.FOR	30
horlen	Length of horizontal section of horizontal well (ft)	Declared in	TYPE8.H	1
		Assigned in	CONVERT.FOR	87, 93, 101
		Called by	CONVLV.FOR	57
		Called by	CONVERT.FOR	94
horspow	Compressor horsepower (HP)	Declared in	RD_DATA.H	16, 32
		Called by	RD_STOR.FOR	13, 22
hp	Compressor horsepower (HP)	Declared in	RD_DATA.H	11, 28
		Assigned in	RD_STOR.FOR	22
iadj	Pointer to adjusted reservoir properties arrays	Declared in	RD_DATA.H	22, 38
		Assigned in	RD_STOR.FOR	154
		Called by	CONVERT.FOR	108, 109, 110
		Called by	RD_STOR.FOR	156, 157, 161, 162, 163, 165, 166, 167
iea	Intangible environmental addback (MM$)	Declared in	CASHFLOW.H	31, 65
		Assigned in	CASHFLOW.FOR	153, 155, 158
		Assigned in	INITCASH.FOR	27
		Called by	CASHFLOW.FOR	171
iexruntyp	Flag of run type for existing storage reservoirs	Declared in	TYPE5.H	10, 12
		Assigned in	STORPERF.FOR	53
		Called by	CONVERT.FOR	107, 118

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Called by	RD_STOR.FOR	155
		Called by	STORPERF.FOR	47, 110, 140, 253, 482
ii	Intangible investment (MM$)	Declared in	CASHFLOW.H	11, 62
		Assigned in	CASHFLOW.FOR	39
		Assigned in	INITCASH.FOR	11
		Assigned in	WRT_PRO.FOR	69, 83, 127
		Called by	CASHFLOW.FOR	58, 94, 105, 107, 179, 223
imod	SRPM module number	Declared in	TYPE10.H	1
		Assigned in	RD_TECH.FOR	60, 62, 64, 66
		Assigned in	MK_TYPE.FOR	49
		Assigned in	CONVERT.FOR	138
		Called by	CONVLV.FOR	47
inj	Injectant costs (MM$)	Declared in	COSTING.H	9, 27
		Assigned in	INITCOST.FOR	11
		Assigned in	WRT_PRO.FOR	63
		Called by	CASHFLOW.FOR	38, 59, 105, 223
intadd	Total intangible addback (MM$)	Declared in	CASHFLOW.H	33, 65
		Assigned in	CASHFLOW.FOR	171
		Assigned in	INITCASH.FOR	29
		Assigned in	WRT_PRO.FOR	137, 159
		Called by	CASHFLOW.FOR	172, 257
intang_m	Intangible multiplier (scalar)	Declared in	UNITCOST.H	25, 38
		Assigned in	INITUNIT.FOR	24
		Assigned in	UNITCOST.FOR	109
		Called by	CASHFLOW.FOR	33, 34
		Called by	PRECOST.FOR	64, 73, 96
io_wells	Number of wells by pay grade	Declared in	RD_DATA.H	12, 32
		Assigned in	CONVERT.FOR	129
		Called by	PRECOST.FOR	55, 59, 60, 65
iowells	Number of wells from storage reservoir database	Declared in	RD_DATA.H	15, 31
		Called by	RD_STOR.FOR	12, 23
ipd	Intangible drilling cost preference deduction (fraction)	Declared in	TAX_NAT.H	14, 50
		Assigned in	RDTAXNAT.FOR	37
		Called by	CASHFLOW.FOR	181
		Called by	RDTAXNAT.FOR	36
ipdr	Independent producer depletion rate (fraction)	Declared in	TAX_NAT.H	6, 50
		Assigned in	RDTAXNAT.FOR	13
		Called by	RDTAXNAT.FOR	12
ira	Maximum AMT reduction for independents (fraction)	Declared in	TAX_NAT.H	16, 50
		Assigned in	RDTAXNAT.FOR	43
		Called by	CASHFLOW.FOR	184

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Called by	RDTAXNAT.FOR	42
jtyp	Well type (0=vertical, 1=horizontal)	Declared in	TYPE8.H	1
		Assigned in	CONVERT.FOR	86, 92, 100
		Called by	CONVLV.FOR	50, 56
		Called by	UNITCOST.FOR	26
kshut	Reservoir shut-in flag (0=on production/injection, 1=shut-in)	Declared in	TYPE9.H	2
		Assigned in	CALCS.FOR	40, 41, 42
		Assigned in	RATE1.FOR	43
		Assigned in	RATE2.FOR	25, 26, 27, 43, 44, 45
		Assigned in	CALCPQ.FOR	25
		Assigned in	CNTRL.FOR	23
		Called by	CALCS.FOR	13, 32
		Called by	RATE1.FOR	14
		Called by	RATE2.FOR	21, 22, 23
la	Lease acquisition costs (MM$)	Declared in	COSTING.H	13, 28
		Assigned in	INITCOST.FOR	15
		Assigned in	PRECOST.FOR	81
		Assigned in	WRT_PRO.FOR	105, 107, 117
		Called by	CASHFLOW.FOR	69, 74, 80, 81, 169, 227, 235, 236, 262
la_oam	Lease acquisition O&M (fixed) costs (MM$)	Declared in	COSTING.H	14, 28
		Assigned in	PRECOST.FOR	83
		Called by	PRECOST.FOR	86, 89
lcst_op1	Levelized investment cost for storage option 1 ($/MCF)	Declared in	STORLP.H	7, 30
		Assigned in	WRT_DI.FOR	27, 44, 51
		Called by	WRT_DI.FOR	31, 56
lcst_op2	Levelized investment cost for storage option 2 ($/MCF)	Declared in	STORLP.H	8, 30
		Assigned in	WRT_DI.FOR	24, 41, 48
		Called by	WRT_DI.FOR	27, 32, 51, 57
lcst_op3	Levelized investment cost for storage option 3 ($/MCF)	Declared in	STORLP.H	9, 30
		Assigned in	WRT_DI.FOR	21, 36
		Called by	WRT_DI.FOR	24, 33, 48, 58
lcst1	Levelized investment cost for storage ($/MCF)	Declared in	STORLP.H	10, 33
		Assigned in	STORPERF.FOR	614, 634
		Called by	WRT_DI.FOR	36, 41, 44
lev_ex	Levelized investment costs for existing storage ($/MCF)	Declared in	STORLP.H	28, 35
		Called by	RDLEVEX.FOR	8
		Called by	WRT_DI.FOR	21
maxdeliv	Maximum deliverability (MMCF/D)	Declared in	RD_DATA.H	10, 27
		Assigned in	RD_STOR.FOR	43
		Called by	CONVERT.FOR	119
		Called by	DATOUT.FOR	63

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Called by	RD_STOR.FOR	14, 42, 44
		Called by	STORPERF.FOR	338, 364
maxdepth	Maximum reservoir depth (ft)	Declared in	RD_DATA.H	8, 26
		Assigned in	RD_STOR.FOR	51, 52
		Called by	RD_STOR.FOR	11, 49, 50
mbgc	Maximum total base gas capacity (MMCF)	Declared in	STORLP.H	6, 30
		Assigned in	DATOUT.FOR	26, 33
		Assigned in	STORPERF.FOR	534, 541, 545
		Called by	DATOUT.FOR	17, 61, 72
		Called by	WRT_DI.FOR	55
		Called by	STORPERF.FOR	543
mecp_op1_1	Maximum extraction capacity of option 1 in season 1 (%)	Declared in	STORLP.H	20, 32
		Assigned in	STORPERF.FOR	574, 582
		Called by	WRT_DI.FOR	31, 39, 40, 56
mecp_op1_2	Maximum extraction capacity of option 1 in season 2 (%)	Declared in	STORLP.H	21, 32
		Assigned in	STORPERF.FOR	575, 583
		Called by	WRT_DI.FOR	31, 39, 56
mecp_op1_3	Maximum extraction capacity of option 1 in season 3 (%)	Declared in	STORLP.H	22, 32
		Assigned in	STORPERF.FOR	576, 584
		Called by	WRT_DI.FOR	32, 40, 57
mecp_op2_1	Maximum extraction capacity of option 2 in season 1 (%)	Declared in	STORLP.H	23, 32
		Assigned in	STORPERF.FOR	577, 585
		Called by	WRT_DI.FOR	33, 58
mecp_op2_2	Maximum extraction capacity of option 2 in season 2 (%)	Declared in	STORLP.H	24, 33
		Assigned in	STORPERF.FOR	578, 586
		Called by	WRT_DI.FOR	33, 58
mecp_op3_1	Maximum extraction capacity of option 3 in season 1 (%)	Declared in	STORLP.H	25, 33
		Assigned in	STORPERF.FOR	579, 587
		Called by	WRT_DI.FOR	34, 59
micp	Maximum injection capacity (%)	Declared in	STORLP.H	26, 33
		Assigned in	STORPERF.FOR	580, 588
		Called by	WRT_DI.FOR	32, 33, 34, 57, 58, 59
min_well	Well spacing (acres)	Declared in	RD_DATA.H	7, 34
		Called by	RD_STOR.FOR	105, 117
		Called by	RD_WSPAC.FOR	12
mindepth	Minimum reservoir depth (ft)	Declared in	RD_DATA.H	8, 26
		Assigned in	RD_STOR.FOR	51, 52
		Called by	RD_STOR.FOR	12, 49, 50
module	GSAM/SRPM module number	Declared in	GSAMVAR.H	42, 55

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Assigned in	CONVERT.FOR	25, 35, 36, 132, 134
		Assigned in	CONVLV.FOR	47
		Assigned in	PRECOST.FOR	70, 71
		Assigned in	RD_STOR.FOR	133
		Assigned in	STORPERF.FOR	619
		Called by	CONVERT.FOR	26, 27, 77, 81, 90, 98, 102, 103, 104, 138
		Called by	CONVLV.FOR	49, 66, 71
		Called by	PD.FOR	1, 19, 25
		Called by	RD_STOR.FOR	32, 33, 38, 90
		Called by	STORPERF.FOR	219
		Declared in	STORLP.H	2
modulesrpm	GSAM/SRPM module number	Declared in	GSAMVAR.H	42, 55
		Assigned in	RD_STOR.FOR	33
		Assigned in	STORPERF.FOR	312, 331, 365, 374, 446, 480
		Called by	CONVERT.FOR	25
		Called by	STORPERF.FOR	168, 476, 619
mwgc	Maximum total working gas capacity (MMCF)	Declared in	STORLP.H	5, 30
		Assigned in	DATOUT.FOR	25, 32
		Assigned in	STORPERF.FOR	269, 273, 304, 308, 344, 348, 397, 401, 436,
				440,501, 504, 533, 540, 544
		Called by	DATOUT.FOR	17, 60, 71
		Called by	PRECOST.FOR	84, 94, 97
		Called by	WRT_DI.FOR	13, 30, 55
		Called by	CASHFLOW.FOR	36, 71, 72, 85, 90, 263
		Called by	STORPERF.FOR	275, 281, 288, 310, 319, 324, 327, 350, 355, 360,
				363, 403, 405, 442, 444, 445, 507, 517, 564, 573,
				574, 575, 576, 577, 578, 579, 580
n_tot_lev	Number of data in LEV.DAT	Declared in	STORLP.H	27, 33
		Assigned in	RDLEVEX.FOR	11
		Called by	WRT_DI.FOR	16, 19
n_tot_reg	Number of data in DWLSPAC.DAT	Declared in	RD_DATA.H	18, 33
		Assigned in	RD_WSPAC.FOR	15
		Called by	RD_STOR.FOR	100, 112
n2	Concentration of N2 (fraction)	Declared in	GSAMVAR.H	76, 140
		Assigned in	RD_STOR.FOR	67
		Assigned in	STORPERF.FOR	234, 238
		Called by	CONVERT.FOR	33
		Called by	STORPERF.FOR	226
nafe	Number of AFE components	Declared in	UNITCOST.H	30, 37
		Assigned in	RD_AFE.FOR	8, 10, 11, 14
		Called by	RD_AFE.FOR	9
narray	Size of pressure function arrays	Declared in	TYPE2.H	1
		Assigned in	PSI.FOR	10
		Assigned in	SETUP.FOR	11, 13

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Assigned in	SETVAR.FOR	7
		Assigned in	TYP_CRV.FOR	22
		Assigned in	VISG.FOR	10
		Assigned in	ZEE.FOR	10
		Called by	CALCPQ.FOR	17, 19, 21, 41, 45, 55, 77, 78
		Called by	CONVLV.FOR	14, 15, 23, 24, 27, 28
		Called by	PRESUR.FOR	1, 5, 6, 11, 12
		Called by	PSI.FOR	1, 5, 6
		Called by	PWELL.FOR	73, 74, 88, 108, 109, 134, 135, 159, 161
		Called by	RATE1.FOR	22, 23, 28, 37, 39, 49, 73, 75, 78, 79
		Called by	REALGS.FOR	7, 8, 17
		Called by	SETUP.FOR	21
		Called by	SETVAR.FOR	1
		Called by	TYP_CRV.FOR	25
		Called by	VISG.FOR	1, 5, 6
		Called by	ZEE.FOR	1, 5, 6
ncis	Number of data in adjusted reservoir property file ROCKPROP.ADJ	Declared in	RD_DATA.H	22, 38
		Assigned in	STORPERF.FOR	119
		Called by	RD_STOR.FOR	156
		Called by	STORPERF.FOR	122
netpay	Net pay thickness (ft)	Declared in	GSAMVAR.H	60, 123
		Assigned in	RD_STOR.FOR	24, 122, 127, 133, 137, 165
		Called by	RD_STOR.FOR	124, 135, 144, 152, 170
		Called by	UNITCOST.FOR	30
		Called by	CONVERT.FOR	43, 126
		Called by	STORPERF.FOR	475
netpay_fac	Geological factor for net pay thickness	Declared in	GEOLOGY.H	7, 11
		Called by	CONVERT.FOR	43
		Called by	RD_GEO.FOR	17, 23
netsales	Net sales (MM$)	Declared in	CASHFLOW.H	7, 61
		Assigned in	CASHFLOW.FOR	37
		Assigned in	INITCASH.FOR	7
		Assigned in	WRT_PRO.FOR	54, 121
		Called by	CASHFLOW.FOR	93, 97, 102, 107
niat	Net income after taxes (MM$)	Declared in	CASHFLOW.H	51, 67
		Assigned in	CASHFLOW.FOR	254
		Assigned in	INITCASH.FOR	47
		Assigned in	WRT_PRO.FOR	149
		Called by	CASHFLOW.FOR	255
nibt	Net income before taxes (MM$)	Declared in	CASHFLOW.H	36, 65
		Assigned in	CASHFLOW.FOR	172
		Assigned in	INITCASH.FOR	32
		Assigned in	WRT_PRO.FOR	141
		Called by	CASHFLOW.FOR	173, 176, 178, 200, 254

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
nibta	Net income before tax addback (MM$)	Declared in	CASHFLOW.H	35, 65
		Assigned in	CASHFLOW.FOR	106
		Assigned in	INITCASH.FOR	31
		Assigned in	WRT_PRO.FOR	133
		Called by	CASHFLOW.FOR	172
nifoag	Net income from oil and gas (MM$)	Declared in	CASHFLOW.H	46, 66
		Assigned in	CASHFLOW.FOR	180, 181
		Assigned in	INITCASH.FOR	42
nil	Logical flag for net income limitations	Declared in	TAX_NAT.H	19, 54
		Assigned in	RDTAXNAT.FOR	52
		Called by	CASHFLOW.FOR	95
nilb	Net income limitation base (MM$)	Declared in	CASHFLOW.H	27, 64
		Assigned in	CASHFLOW.FOR	93
		Assigned in	INITCASH.FOR	23
		Called by	CASHFLOW.FOR	96, 97
nill	Net income limitation limit (fraction)	Declared in	TAX_NAT.H	20, 50
		Assigned in	RDTAXNAT.FOR	55
		Called by	CASHFLOW.FOR	97
		Called by	RDTAXNAT.FOR	54
nreg	Number of storage reservoir database in REGIONS.DAT	Declared in	GLOBAL.H	8, 17
		Assigned in	RD_REGS.FOR	26
		Called by	STORPERF.FOR	130
nrestype	Number of reservoir types	Declared in	GEOLOGY.H	3, 13
		Called by	CONVERT.FOR	27
		Called by	RD_GEO.FOR	6, 7, 14
ntax_st	Number of tax regions	Declared in	TAX_REG.H	5, 17
		Called by	CASHFLOW.FOR	25
		Called by	RD_TAX.FOR	9, 10, 13
		Called by	WRT_PRO.FOR	18
nwell	Number of wells	Declared in	FIELD.H	14, 25
		Assigned in	STORPERF.FOR	621
		Called by	PRECOST.FOR	55, 56, 60, 112, 113, 115, 118, 119, 121
nyr	Number of years for potential storage run (years)	Declared in	GLOBAL.H	7, 17
		Assigned in	CASHFLOW.FOR	21
		Assigned in	WRT_TCP.FOR	14
		Called by	CASHFLOW.FOR	30, 84, 88
		Called by	PRECOST.FOR	110
		Called by	WRT_PRO.FOR	21, 28
		Called by	WRT_TCP.FOR	18, 22, 26, 30
nyrset	Number of years for potential storage run (years)	Declared in	GLOBAL.H	7, 17
		Assigned in	RD_STOR.FOR	63

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
nyrset_storage	Number of years for potential storage run (years)	Declared in	GLOBAL.H	7, 17
		Called by	CASHFLOW.FOR	21
		Called by	PRECOST.FOR	66, 72, 78, 82, 114, 120
		Called by	STORPERF.FOR	49, 563, 622
oam	Total O&M costs (MM$)	Declared in	COSTING.H	8, 27
		Assigned in	INITCOST.FOR	10
		Assigned in	PRECOST.FOR	51, 86
		Assigned in	WRT_PRO.FOR	64
		Called by	CASHFLOW.FOR	38, 59
oam_gas	Gas O&M costs ($/MCF)	Declared in	COST.H	10, 49
		Called by	RD_COST.FOR	88
		Called by	UNITCOST.FOR	98
oam_h2o	Surface H2O O&M costs ($/BBL)	Declared in	COST.H	8, 49
		Called by	RD_COST.FOR	86
		Called by	UNITCOST.FOR	97
oam_inc	O&M costs, incremental per 1000 feet ($/MCF)	Declared in	COST.H	11, 49
		Called by	RD_COST.FOR	88
		Called by	UNITCOST.FOR	98
oam_m	O&M costs multiplier (scalar)	Declared in	UNITCOST.H	26, 38
		Assigned in	INITUNIT.FOR	25
		Assigned in	UNITCOST.FOR	110
		Called by	PRECOST.FOR	87, 89, 91, 94
ogip1	Original gas in place (MMCF)	Declared in	TYPE3.H	2
		Assigned in	SETUP.FOR	22
		Called by	RATE1.FOR	41
		Called by	CALCPQ.FOR	24, 58
		Called by	DATOUT.FOR	30
		Called by	GET_TYPE.FOR	27
		Called by	STORPERF.FOR	538
oia	Other intangible addbacks (MM$)	Declared in	CASHFLOW.H	30, 64
		Assigned in	CASHFLOW.FOR	135, 137, 140, 144, 146, 149
		Assigned in	INITCASH.FOR	26
		Called by	CASHFLOW.FOR	171
otc	Other tangible capital costs (MM$)	Declared in	COSTING.H	15, 28
		Assigned in	INITCOST.FOR	18
		Assigned in	PRECOST.FOR	95
		Assigned in	WRT_PRO.FOR	93
		Called by	CASHFLOW.FOR	45, 261
pay	Pay thickness from storage reservoir database (ft)	Declared in	RD_DATA.H	8, 26
		Called by	RD_STOR.FOR	11, 24

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
payadj	Adjusted net pay thickness (ft)	Declared in	RD_DATA.H	19, 36
		Called by	RD_STOR.FOR	165
		Called by	STORPERF.FOR	115
pdr	Depletion rate (fraction)	Declared in	TAX_NAT.H	21, 50
		Assigned in	RDTAXNAT.FOR	58
		Called by	CASHFLOW.FOR	97, 102
		Called by	RDTAXNAT.FOR	57
peakrate	Peak production/injection rate (MCF/day/well)	Declared in	FIELD.H	15, 26
		Assigned in	DATOUT.FOR	49
		Assigned in	STORPERF.FOR	469, 611
		Called by	DATOUT.FOR	63, 74
		Called by	UNITCOST.FOR	66, 69, 72
		Called by	STORPERF.FOR	115, 479
peakrate1	Peak withdrawal rate (MCF/day/well)	Declared in	FIELD.H	16, 27
		Assigned in	STORPERF.FOR	604
		Called by	UNITCOST.FOR	32, 33
		Called by	STORPERF.FOR	611
peakrate2	Peak injection rate (MCF/day/well)	Declared in	FIELD.H	17, 27
		Assigned in	STORPERF.FOR	610
		Called by	UNITCOST.FOR	32, 36
		Called by	STORPERF.FOR	611
perhor	Horizontal direction permeability (md)	Declared in	GSAMVAR.H	63, 124
		Assigned in	RD_STOR.FOR	175
		Assigned in	STORPERF.FOR	386, 408, 409, 425
		Called by	CONVERT.FOR	39
		Called by	STORPERF.FOR	385, 387, 388, 404, 422, 426, 427, 443, 472
perm	Horizontal direction permeability (md)	Declared in	TYPE3.H	1
		Assigned in	RD_STOR.FOR	95, 161
		Assigned in	INIT_WEL.FOR	32
		Assigned in	CONVERT.FOR	39
		Called by	CONVLV.FOR	36, 38, 52, 60, 68
		Called by	RD_STOR.FOR	170, 175, 176, 177
		Called by	DATOUT.FOR	90
		Called by	MK_TYPE.FOR	32
		Called by	CONVERT.FOR	96
perm_fac	Geological factor for permeability	Declared in	GEOLOGY.H	9, 11
		Called by	CONVERT.FOR	39, 40, 44
		Called by	RD_GEO.FOR	18, 24
permadj	Adjusted horizontal direction permeability (md)	Declared in	RD_DATA.H	19, 36
		Called by	RD_STOR.FOR	161
		Called by	STORPERF.FOR	114
permi	Horizontal direction permeability from storage database (md)	Declared in	RD_DATA.H	11, 28

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Assigned in	RD_STOR.FOR	83, 87, 92
		Called by	RD_STOR.FOR	14, 82, 84, 85, 86, 88, 95, 110
permma	Matrix permeability by pay grade (md)	Declared in	TYPE3.H	2
		Assigned in	INIT_WEL.FOR	38
		Assigned in	CONVERT.FOR	44
		Called by	CONVLV.FOR	68
		Called by	MK_TYPE.FOR	41
permtx	Matrix permeability (md)	Declared in	GSAMVAR.H	65, 124
		Assigned in	RD_STOR.FOR	177
		Assigned in	STORPERF.FOR	388, 427
		Called by	CONVERT.FOR	44
permv	Vertical direction permeability (md)	Declared in	TYPE3.H	1
		Assigned in	INIT_WEL.FOR	33
		Assigned in	CONVERT.FOR	40
		Called by	CONVLV.FOR	60
		Called by	MK_TYPE.FOR	32
		Called by	CONVERT.FOR	96
pervrt	Vertical direction permeability (md)	Declared in	GSAMVAR.H	64, 124
		Assigned in	RD_STOR.FOR	176
		Assigned in	STORPERF.FOR	387, 426
		Called by	CONVERT.FOR	40
pggc	Rate of G&G as tangible (depletable) (fraction)	Declared in	TAX_REG.H	13, 19
		Assigned in	RDTAXNAT.FOR	142
		Assigned in	WRT_PRO.FOR	105, 109, 119
		Called by	CASHFLOW.FOR	69, 74, 76, 77, 166, 226, 233, 234
		Called by	RDTAXNAT.FOR	141
piic	Fraction of intangible investment to capital (fraction)	Declared in	TAX_NAT.H	22, 50
		Assigned in	RDTAXNAT.FOR	61
		Called by	CASHFLOW.FOR	42, 43, 44, 112, 123, 135, 144, 153
		Called by	RDTAXNAT.FOR	60
pinit	Initial reservoir pressure (psia)	Declared in	TYPE3.H	1
		Assigned in	CALCS.FOR	56
		Assigned in	INIT_WEL.FOR	31
		Assigned in	CONVERT.FOR	38
		Called by	CALCS.FOR	33
		Called by	CONVLV.FOR	13
		Called by	RATE1.FOR	23, 36
		Called by	RATE2.FOR	14
		Called by	CALCPQ.FOR	16, 40, 54, 58, 62, 86
		Called by	MK_TYPE.FOR	32
		Called by	CNTRL.FOR	21
		Called by	SETUP.FOR	15, 20
pl	Langmuir pressure (not currently used) (psia)	Declared in	TYPE9.H	4

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Assigned in	INIT_WEL.FOR	44
		Called by	MK_TYPE.FOR	67
plac	Fraction lease acquisition costs tangible (fraction)	Declared in	TAX_REG.H	16, 19
		Assigned in	RDTAXNAT.FOR	150
		Assigned in	WRT_PRO.FOR	105, 107, 117
		Called by	CASHFLOW.FOR	69, 74, 80, 81, 169, 227, 235, 236
		Called by	RDTAXNAT.FOR	149
pmin	Minimum allowable wellhead pressure (psia)	Declared in	TYPE5.H	1, 5
		Assigned in	CNTRL.FOR	15
		Assigned in	SETVAR.FOR	12
por_fac	Geological factor for porosity	Declared in	GEOLOGY.H	6, 11
		Called by	CONVERT.FOR	41, 45
		Called by	RD_GEO.FOR	17, 23
poradj	Adjusted porosity (fraction)	Declared in	RD_DATA.H	19, 36
		Called by	RD_STOR.FOR	162
		Called by	STORPERF.FOR	114
pori	Porosity from storage reservoir database (%)	Declared in	RD_DATA.H	11, 28
		Assigned in	RD_STOR.FOR	85, 88, 91
		Called by	RD_STOR.FOR	14, 82, 83, 84, 86, 94
porma	Matrix porosity by pay grade (fraction)	Declared in	TYPE3.H	2
		Assigned in	INIT_WEL.FOR	39
		Assigned in	CONVERT.FOR	45
		Called by	CONVLV.FOR	67
		Called by	MK_TYPE.FOR	41
pormtx	Matrix porosity (fraction)	Declared in	GSAMVAR.H	67, 125
		Assigned in	RD_STOR.FOR	179
		Called by	CONVERT.FOR	45
poros	Porosity by pay grade (fraction)	Declared in	TYPE3.H	1
		Assigned in	INIT_WEL.FOR	34
		Assigned in	CONVERT.FOR	41
		Called by	CONVLV.FOR	32, 37, 67
		Called by	CALCPQ.FOR	53
		Called by	DATOUT.FOR	89
		Called by	MK_TYPE.FOR	33
		Called by	CONVERT.FOR	68, 71, 72, 73
		Called by	SETUP.FOR	22
portot	Total porosity (fraction)	Declared in	GSAMVAR.H	66, 125
		Assigned in	RD_STOR.FOR	178
		Called by	RD_STOR.FOR	179
		Called by	CONVERT.FOR	41
		Called by	STORPERF.FOR	473

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
prbh	Bottomhole pressure (psia)	Declared in	TYPE5.H	2, 6
		Assigned in	CALCPQ.FOR	49
		Assigned in	WRT_TCP.FOR	18
		Assigned in	SETVAR.FOR	15
		Called by	DATOUT.FOR	105
		Called by	GET_TYPE.FOR	36
preary	Pressure array (psia)	Declared in	TYPE2.H	1
		Assigned in	REALGS.FOR	8
		Called by	CALCPQ.FOR	17, 19, 21, 41, 45, 55, 60, 84, 98, 99
		Called by	CONVLV.FOR	14, 15, 23, 24, 27, 28
		Called by	PRESUR.FOR	1, 2, 6, 8, 26
		Called by	PSI.FOR	1, 2, 5, 8, 13, 14
		Called by	PWELL.FOR	73, 74, 88, 108, 109, 134, 135, 159, 161
		Called by	RATE1.FOR	22, 23, 28, 37, 39, 49, 73, 75, 78, 79
		Called by	REALGS.FOR	16, 18, 23
		Called by	SETUP.FOR	21
		Called by	VISG.FOR	1, 2, 5, 8, 13
		Called by	ZEE.FOR	1, 2, 5, 8, 13
preavg	Average reservoir pressure (psia)	Declared in	TYPE5.H	3, 7
		Assigned in	CALCS.FOR	15, 35, 65
		Assigned in	CONVLV.FOR	19
		Assigned in	CALCPQ.FOR	71, 73, 102
		Assigned in	CNTRL.FOR	20, 21
		Called by	CALCS.FOR	34, 47, 48, 49, 51, 52, 53
		Called by	CONVLV.FOR	18, 21
		Called by	RATE2.FOR	16, 17
		Called by	CALCPQ.FOR	20
premin	Minimum allowable wellhead pressure (psia)	Declared in	TYPE7.H	1
		Assigned in	CONVERT.FOR	124, 125
		Called by	RATE1.FOR	38
		Called by	RATE2.FOR	18
		Called by	SOLVER.FOR	30
		Called by	CALCPQ.FOR	18
		Called by	MK_TYPE.FOR	74
		Called by	UNITCOST.FOR	25
		Called by	CNTRL.FOR	15
presin	Initial reservoir pressure (psia)	Declared in	GSAMVAR.H	71, 126
		Assigned in	RD_STOR.FOR	61
		Called by	RD_STOR.FOR	123
		Called by	UNITCOST.FOR	36
		Called by	CONVERT.FOR	38
pressure	Reservoir pressure (psia)	Declared in	RD_DATA.H	8, 26
		Assigned in	RD_STOR.FOR	56
		Called by	RD_STOR.FOR	12, 53, 54, 55, 57, 61
prod_period	Production period (currently set to 121/365 years) (years)	Declared in	TYPE5.H	9, 12

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Assigned in	STORPERF.FOR	55, 61
		Called by	TYP_CRV.FOR	24
		Called by	RD_STOR.FOR	43
		Called by	STORPERF.FOR	562, 565, 608
prorat_tech	Proration factor to adjust "ratmax"	Declared in	TECH.H	6, 18
		Called by	CONVERT.FOR	121, 122
		Called by	RD_TECH.FOR	27, 30
prwh	Wellhead pressure (psia)	Declared in	TYPE5.H	2, 6
		Assigned in	CALCPQ.FOR	50
		Assigned in	WRT_TCP.FOR	22
		Assigned in	SETVAR.FOR	14
		Called by	DATOUT.FOR	105
		Called by	GET_TYPE.FOR	37
psiary	Pseudo-pressure array (psia^2/cp)	Declared in	TYPE2.H	1
		Assigned in	REALGS.FOR	9, 29
		Called by	CALCPQ.FOR	41, 45
		Called by	CONVLV.FOR	15, 24, 27
		Called by	PRESUR.FOR	1, 2, 5, 7, 8, 18, 25
		Called by	PSI.FOR	1, 2, 6, 12, 16
		Called by	RATE1.FOR	22, 23, 28, 49, 73, 75
psicon	Constant term of dimensionless flow rate	Declared in	TYPE6.H	1
		Assigned in	CONVLV.FOR	38
		Called by	CALCPQ.FOR	44
		Called by	CONVLV.FOR	75
		Called by	RATE1.FOR	26, 33, 54, 72, 80
psys_tech	Minimum allowable wellhead pressure (psia)	Declared in	TECH.H	10, 19
		Called by	CONVERT.FOR	124, 125
		Called by	RD_TECH.FOR	52, 55
qg	Gas production rate (MCF/D/well)	Declared in	TYPE5.H	2, 6
		Assigned in	RATE1.FOR	15, 96
		Assigned in	CNTRL.FOR	16
		Assigned in	SETVAR.FOR	16
		Called by	CALCS.FOR	64
		Called by	RATE1.FOR	26, 101
		Called by	SOLVER.FOR	19
		Called by	CALCPQ.FOR	23, 30, 31, 33, 35, 42
		Called by	DATOUT.FOR	41, 42, 43, 100
		Called by	GET_TYPE.FOR	31
		Called by	STORPERF.FOR	272, 307, 347, 400, 439, 461, 462, 463, 554, 596,
				597, 598
qmax	Maximum gas flow rate (MCF/D/well)	Declared in	TYPE5.H	1, 5
		Assigned in	SETVAR.FOR	13
qreg	Maximum allowable number of regions	Declared in	DIMEN.H	4, 16

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Called by	CONVERT.FOR	125
		Called by	RD_COST.FOR	49, 50, 56, 115, 116, 119, 120, 121, 122
		Called by	RD_TECH.FOR	11, 46, 55, 82
		Called by	STORPERF.FOR	560
		Called by	UNITCOST.FOR	42, 77
qrestype	Maximum allowable number of reservoir types	Declared in	DIMEN.H	13, 25
		Called by	CONVERT.FOR	28
		Called by	RD_GEO.FOR	7, 22, 23, 24
qstate	Maximum allowable number of states	Declared in	DIMEN.H	14, 26
		Called by	CASHFLOW.FOR	26
		Called by	CONVERT.FOR	122
		Called by	RD_TAX.FOR	21
		Called by	RD_TECH.FOR	30
qyr	Maximum allowable number of years for potential storage run	Declared in	DIMEN.H	3, 15
		Called by	CASHFLOW.FOR	16, 27, 64
		Called by	INITCASH.FOR	5
		Called by	INITCOST.FOR	5
		Called by	INITUNIT.FOR	22
		Called by	INIT_WEL.FOR	18
		Called by	PRECOST.FOR	31, 32, 33, 34, 35, 43
		Called by	UNITCOST.FOR	107
		Called by	WRT_PRO.FOR	17, 20
ratmax	Maximum total allowable gas flow rate (MMCF/D)	Declared in	TYPE7.H	1
		Assigned in	SOLVER.FOR	25, 27
		Assigned in	CONVERT.FOR	119, 121, 122
		Assigned in	STORPERF.FOR	266, 295, 336, 391, 430, 499
		Called by	RATE2.FOR	31, 48, 55
		Called by	SOLVER.FOR	24, 26, 33
		Called by	MK_TYPE.FOR	74
regname	Region name	Declared in	RD_DATA.H	5, 33
		Called by	RD_STOR.FOR	100, 112
		Called by	RD_WSPAC.FOR	12
regnm	Prefix of file name of storage reservoir database	Declared in	GLOBAL.H	13, 21
		Assigned in	STORPERF.FOR	132
		Called by	RD_REGS.FOR	22
		Called by	STORPERF.FOR	131, 137, 142, 155, 162, 170, 172, 177, 180,
				186, 189, 194, 197, 203, 206, 212, 214, 220,
				222, 229, 411, 413, 513
res_map	Reservoir type	Declared in	GEOLOGY.H	4, 13
		Called by	CONVERT.FOR	27
		Called by	RD_GEO.FOR	16
rescod	3-Digit reservoir code from storage reservoir database	Declared in	GSAMVAR.H	14, 45
		Called by	RD_STOR.FOR	35

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
resid	Storage ID	Declared in	RD_DATA.H	4, 25
		Called by	DATOUT.FOR	50
		Called by	RD_STOR.FOR	10, 20
resv	30-Character reservoir name from storage reservoir database	Declared in	RD_DATA.H	2, 25
		Called by	DATOUT.FOR	52
		Called by	RD_STOR.FOR	10, 30
rhoma	Matrix density (not currently used)	Declared in	TYPE9.H	4
		Assigned in	INIT_WEL.FOR	47
		Called by	MK_TYPE.FOR	67
royrate	Royalty rate (fraction)	Declared in	TYPE4.H	1
		Assigned in	CONVERT.FOR	84
		Assigned in	MK_TYPE.FOR	50
		Called by	CONVLV.FOR	48, 61
rw	Wellbore radius (ft)	Declared in	TYPE4.H	1
		Assigned in	MK_TYPE.FOR	50
		Assigned in	CONVERT.FOR	84
		Called by	CONVLV.FOR	48, 61
salin	Water salinity (ppm by weight)	Declared in	TYPE3.H	2
		Assigned in	INIT_WEL.FOR	37
		Called by	CALCPQ.FOR	63, 87
		Called by	CONVLV.FOR	31
		Called by	MK_TYPE.FOR	33
		Called by	PWELL.FOR	14, 18, 19, 20, 21, 29, 33, 34, 35, 36, 48
sevtax	Severance tax (MM$)	Declared in	CASHFLOW.H	55, 67
		Assigned in	CASHFLOW.FOR	70, 73
		Assigned in	INITCASH.FOR	51
		Assigned in	WRT_PRO.FOR	123
		Called by	CASHFLOW.FOR	93, 107
sfit	Selected federal income taxes (MM$)	Declared in	CASHFLOW.H	57, 67
		Assigned in	CASHFLOW.FOR	202, 204
		Assigned in	INITCASH.FOR	53
		Called by	CASHFLOW.FOR	216
sgadj	Adjusted gas saturation (fraction)	Declared in	RD_DATA.H	19, 36
		Called by	RD_STOR.FOR	163
		Called by	STORPERF.FOR	114
sgas	Gas saturation (fraction)	Declared in	RD_DATA.H	11, 28
		Called by	RD_STOR.FOR	15, 27
skin	Skin factor	Declared in	TYPE5.H	2, 6
		Assigned in	CONVERT.FOR	113, 114, 115, 116
		Assigned in	STORPERF.FOR	300, 338, 393, 432

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Called by	RATE1.FOR	17
		Called by	CALCPQ.FOR	43
		Called by	MK_TYPE.FOR	75, 79, 81
		Called by	STORPERF.FOR	278, 285, 341
skinadj	Adjusted skin factor	Declared in	RD_DATA.H	20, 37
		Called by	CONVERT.FOR	108, 109, 110
		Called by	STORPERF.FOR	115
soil	Oil saturation (fraction)	Declared in	RD_DATA.H	11, 28
		Called by	RD_STOR.FOR	14, 28
state	4-digit state ID	Declared in	GSAMVAR.H	15, 45
		Assigned in	RD_STOR.FOR	21
		Called by	CASHFLOW.FOR	25
		Called by	UNITCOST.FOR	46
		Called by	WRT_PRO.FOR	18
statin	Flag for type of storage reservoir (0=existing, 1=potential)	Declared in	GSAMVAR.H	21, 46
		Assigned in	RD_STOR.FOR	36, 37
		Called by	CONVERT.FOR	107, 118
		Called by	DATOUT.FOR	53
		Called by	RD_STOR.FOR	34, 44, 110, 155
		Called by	STORPERF.FOR	253, 482, 486, 590
stid	4-digit state ID from storage reservoir database	Declared in	RD_DATA.H	16, 32
		Called by	RD_STOR.FOR	11, 21
stim	Stimulation cost (MM$)	Declared in	COSTING.H	19, 28
		Assigned in	INITCOST.FOR	19
		Assigned in	PRECOST.FOR	64, 67, 73
		Assigned in	WRT_PRO.FOR	66
		Called by	CASHFLOW.FOR	60
		Called by	PRECOST.FOR	88
stim_w	Stimulation unit cost (MM$/well)	Declared in	UNITCOST.H	10, 33
		Assigned in	INITUNIT.FOR	7
		Assigned in	UNITCOST.FOR	27
		Called by	PRECOST.FOR	64, 73
		Called by	UNITCOST.FOR	29, 65
stimfac	Stimulation cost based on stimulation length (MM$)	Declared in	COST.H	9, 49
		Called by	RD_COST.FOR	84
		Called by	UNITCOST.FOR	30
stor_fact	Cost of injection gas as a fraction of gas price (fraction)	Declared in	TYPE5.H	9, 12
		Assigned in	STORPERF.FOR	57
stor_gas_cost	Cost of base gas (not calculated in SRPM) (MM$)	Declared in	COSTING.H	25, 29
		Assigned in	PRECOST.FOR	50, 77
		Assigned in	WRT_PRO.FOR	75

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Called by	CASHFLOW.FOR	40, 261
strate	State income tax rate (fraction)	Declared in	TAX_REG.H	7, 18
		Assigned in	RD_TAX.FOR	15, 22
		Called by	CASHFLOW.FOR	176
stt	2-Character state code from storage reservoir database	Declared in	RD_DATA.H	3, 25
		Called by	RD_STOR.FOR	11
sttax	State income taxes (MM$)	Declared in	CASHFLOW.H	37, 65
		Assigned in	CASHFLOW.FOR	174, 176
		Assigned in	INITCASH.FOR	33
		Assigned in	WRT_PRO.FOR	143
		Called by	CASHFLOW.FOR	178, 200, 254
swat	Water saturation (fraction)	Declared in	RD_DATA.H	11, 28
		Called by	RD_STOR.FOR	15, 26
swi	Initial water saturation by pay grade (fraction)	Declared in	TYPE3.H	1
		Assigned in	CONVERT.FOR	42, 50
		Assigned in	INIT_WEL.FOR	35
		Called by	CALCPQ.FOR	64, 88, 97
		Called by	CONVERT.FOR	68, 71, 72, 73
		Called by	CONVLV.FOR	33, 34
		Called by	MK_TYPE.FOR	33
		Called by	SETUP.FOR	23
tang_dwc	Tangible development cost (MM$)	Declared in	CASHFLOW.H	13, 64
		Assigned in	CASHFLOW.FOR	31
		Assigned in	INITCASH.FOR	60
		Assigned in	WRT_PRO.FOR	89
		Called by	CASHFLOW.FOR	45, 50, 53, 228, 237, 261
tang_ewc	Tangible exploratory cost (MM$)	Declared in	CASHFLOW.H	15, 64
		Assigned in	CASHFLOW.FOR	32
		Assigned in	INITCASH.FOR	59
		Assigned in	WRT_PRO.FOR	87
		Called by	CASHFLOW.FOR	45, 50, 53, 229, 238
tang_m	Tangible multiplier (scalar)	Declared in	UNITCOST.H	24, 38
		Assigned in	INITUNIT.FOR	23
		Assigned in	UNITCOST.FOR	108
		Called by	CASHFLOW.FOR	31, 32
		Called by	PRECOST.FOR	75, 95, 107, 108
tax_st	Region identifier	Declared in	TAX_REG.H	6, 17
		Called by	CASHFLOW.FOR	25
		Called by	RD_TAX.FOR	14, 21
		Called by	WRT_PRO.FOR	18
tci	Total capitalized investments (MM$)	Declared in	CASHFLOW.H	18, 62

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Assigned in	CASHFLOW.FOR	46
		Assigned in	INITCASH.FOR	14
		Assigned in	WRT_PRO.FOR	97
		Called by	CASHFLOW.FOR	47, 57
		Called by	STORPERF.FOR	248
tciadj	Adjusted total capitalized investments (MM$)	Declared in	CASHFLOW.H	19, 63
		Assigned in	CASHFLOW.FOR	47, 49, 52, 54
		Assigned in	INITCASH.FOR	15
		Assigned in	WRT_PRO.FOR	99
		Called by	CASHFLOW.FOR	57
tdes	Sorption time constant (not currently used)	Declared in	TYPE9.H	4
		Assigned in	INIT_WEL.FOR	45
		Called by	MK_TYPE.FOR	67
tdtcr	Tangible development tax credit rate (fraction)	Declared in	TAX_NAT.H	33, 52
		Assigned in	RDTAXNAT.FOR	94
		Called by	CASHFLOW.FOR	50, 53, 228, 237
		Called by	RDTAXNAT.FOR	93
tem	Bottomhole temperature (deg. F)	Declared in	TYPE1.H	2
		Assigned in	CONVERT.FOR	30
		Called by	CONVLV.FOR	31, 38
		Called by	PWELL.FOR	14, 18, 19, 20, 21, 29, 33, 34, 35, 36, 48, 77, 80,
				89, 93, 112, 116, 119, 136, 141, 163
		Called by	CALCPQ.FOR	63, 87
		Called by	MK_TYPE.FOR	25
		Called by	SETUP.FOR	22
		Called by	RHOW.FOR	1, 4, 5
		Called by	CWATER.FOR	1, 2, 13
		Called by	REALGS.FOR	5, 6
		Called by	BW.FOR	1, 2, 3
		Called by	VISGA.FOR	1, 4
		Called by	VISW.FOR	1, 5, 7, 8
tfit	Tenative federal income taxes (MM$)	Declared in	CASHFLOW.H	56, 67
		Assigned in	CASHFLOW.FOR	200
		Assigned in	INITCASH.FOR	52
		Called by	CASHFLOW.FOR	202, 204, 211, 212, 218, 220
thick	Pay thickness (ft)	Declared in	TYPE3.H	2
		Assigned in	INIT_WEL.FOR	36
		Assigned in	CONVERT.FOR	43, 73
		Called by	CONVLV.FOR	38, 61, 62
		Called by	DATOUT.FOR	88
		Called by	MK_TYPE.FOR	33
		Called by	CONVERT.FOR	67, 71, 72, 97
		Called by	SETUP.FOR	22
ti	Tangible investments (MM$)	Declared in	CASHFLOW.H	17, 62

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Assigned in	CASHFLOW.FOR	45
		Assigned in	INITCASH.FOR	13
		Assigned in	WRT_PRO.FOR	83, 85, 157
		Called by	CASHFLOW.FOR	46, 58, 223, 256
timcon	Constant term of dimensionless time	Declared in	TYPE6.H	1
		Assigned in	CONVLV.FOR	36
		Called by	CONVLV.FOR	45
time	Time grid (years)	Declared in	TYPE_OUT.H	4, 6, 8
		Assigned in	CNTRL.FOR	10
		Called by	CONVLV.FOR	43, 44
		Called by	RATE1.FOR	34, 35
		Called by	CALCPQ.FOR	13, 14, 31, 35
		Called by	DATOUT.FOR	38, 39, 40, 99, 104
		Called by	STORPERF.FOR	458, 459, 460, 593, 594, 595
		Declared in	TYPE5.H	1, 5
toc	Total operating cost (MM$)	Declared in	CASHFLOW.H	8, 61
		Assigned in	CASHFLOW.FOR	59
		Assigned in	INITCASH.FOR	8
		Assigned in	WRT_PRO.FOR	55, 125
		Called by	CASHFLOW.FOR	93, 107
		Called by	WRT_PRO.FOR	22
togip	Original gas in place (MMCF)	Declared in	RD_DATA.H	11, 28
		Assigned in	RD_STOR.FOR	124, 132, 144, 152
		Called by	RD_STOR.FOR	125, 126, 127, 128, 129, 145
		Called by	DATOUT.FOR	59, 70
		Called by	STORPERF.FOR	488, 507, 517, 544, 545
totbase	Total base gas capacity from storage reservoir database (MMCF)	Declared in	RD_DATA.H	10, 27
		Called by	DATOUT.FOR	61
		Called by	RD_STOR.FOR	13
totfutr	Total future unused capacity from storage reservoir database (MMCF)	Declared in	RD_DATA.H	10, 27
		Called by	RD_STOR.FOR	14
totwork	Total working gas capacity (MMCF)	Declared in	RD_DATA.H	10, 27
		Called by	DATOUT.FOR	60
		Called by	RD_STOR.FOR	13, 42, 43
		Called by	STORPERF.FOR	275, 310, 311, 350, 403, 442
transcst	Transportation cost (MM$)	Declared in	COSTING.H	11, 28
		Assigned in	INITCOST.FOR	13
		Assigned in	WRT_PRO.FOR	50
		Called by	CASHFLOW.FOR	36
type_base	Base gas capacity by pay grade (MMCF)	Declared in	TYPE_OUT.H	15, 25
		Assigned in	DATOUT.FOR	30

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Assigned in	INIT_WEL.FOR	24
		Assigned in	STORPERF.FOR	538
		Called by	DATOUT.FOR	33
		Called by	STORPERF.FOR	541
type_gas	Gas production (BCF)	Declared in	TYPE_OUT.H	5, 9, 23
		Assigned in	INIT_WEL.FOR	19
		Assigned in	GET_TYPE.FOR	30
type_ogip	OGIP by pay grade (BCF)	Declared in	TYPE_OUT.H	14, 25
		Assigned in	INIT_WEL.FOR	27
		Assigned in	GET_TYPE.FOR	27
type_pbhp	Bottomhole pressure (psia)	Declared in	TYPE_OUT.H	11, 23
		Assigned in	INIT_WEL.FOR	20
		Assigned in	GET_TYPE.FOR	36
type_pwhp	Wellhead pressure (psia)	Declared in	TYPE_OUT.H	12, 23
		Assigned in	INIT_WEL.FOR	21
		Assigned in	GET_TYPE.FOR	37
type_well	Number of wells by pay grade	Declared in	TYPE_OUT.H	10, 25
		Assigned in	INIT_WEL.FOR	26
		Assigned in	GET_TYPE.FOR	28
		Called by	DATOUT.FOR	29, 30, 47, 85, 100, 101, 102
		Called by	STORPERF.FOR	272, 273, 307, 308, 347, 348, 400, 401, 439, 440,
				467, 504, 537, 538, 554, 556, 621
type_work	Working gas capacity by pay grade (MMCF)	Declared in	TYPE_OUT.H	16, 25
		Assigned in	DATOUT.FOR	29
		Assigned in	INIT_WEL.FOR	25
		Assigned in	STORPERF.FOR	537
		Called by	DATOUT.FOR	31, 32
		Called by	STORPERF.FOR	539, 540, 562
uamti	Unadjusted AMT income (MM$)	Declared in	CASHFLOW.H	44, 66
		Assigned in	CASHFLOW.FOR	184, 186
		Assigned in	INITCASH.FOR	40
		Called by	CASHFLOW.FOR	193, 194, 195, 197
ucpamt	Useable credits for past AMT (MM$)	Declared in	CASHFLOW.H	58, 67
		Assigned in	CASHFLOW.FOR	212, 214
		Assigned in	INITCASH.FOR	54
		Called by	CASHFLOW.FOR	216
va	Gas viscosity at 1 atm (cp)	Declared in	TYPE1.H	2
		Assigned in	REALGS.FOR	6
		Called by	PWELL.FOR	74, 109, 135
		Called by	RATE1.FOR	79
		Called by	REALGS.FOR	22, 27, 30
		Called by	VISG.FOR	1, 4, 15

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
var_ex	Variable O&M costs for existing storage ($/MCF)	Declared in	STORLP.H	28, 35
		Called by	RDLEVEX.FOR	8
		Called by	WRT_DI.FOR	23
visary	Viscosity array (cp)	Declared in	TYPE2.H	1
		Assigned in	REALGS.FOR	30
		Called by	PWELL.FOR	74, 109, 135
		Called by	RATE1.FOR	79
		Called by	VISG.FOR	1, 2, 6, 12, 15
voam_g	Surface O&M - gas ($/MCF)	Declared in	UNITCOST.H	18, 34
		Assigned in	INITUNIT.FOR	11
		Assigned in	UNITCOST.FOR	98
		Called by	PRECOST.FOR	84
vom	Variable O&M cost for storage ($/MCF)	Declared in	STORLP.H	14, 31
		Assigned in	STORPERF.FOR	613, 630
		Called by	WRT_DI.FOR	38, 43, 46
vom_op1	Variable O&M cost for storage option 1 ($/MCF)	Declared in	STORLP.H	11, 31
		Assigned in	WRT_DI.FOR	29, 46, 53
		Called by	WRT_DI.FOR	31, 56
vom_op2	Variable O&M cost for storage option 2 ($/MCF)	Declared in	STORLP.H	12, 31
		Assigned in	WRT_DI.FOR	26, 43, 50
		Called by	WRT_DI.FOR	29, 32, 53, 57
vom_op3	Variable O&M cost for storage option 3 ($/MCF)	Declared in	STORLP.H	13, 31
		Assigned in	WRT_DI.FOR	23, 38
		Called by	WRT_DI.FOR	26, 34, 50, 59
watsat	Water saturation (fraction)	Declared in	GSAMVAR.H	69, 126
		Assigned in	RD_STOR.FOR	26, 72, 76, 80, 130, 151, 164
		Called by	RD_STOR.FOR	69, 78, 83, 85, 88
		Called by	CONVERT.FOR	42
welrad	Wellbore radius (ft)	Declared in	GSAMVAR.H	78, 126
		Assigned in	RD_STOR.FOR	180
		Assigned in	CONVERT.FOR	81
		Called by	CONVERT.FOR	84
wlspac	Well spacing (acres)	Declared in	GSAMVAR.H	99, 129
		Assigned in	RD_STOR.FOR	105, 109, 117, 131, 139, 167

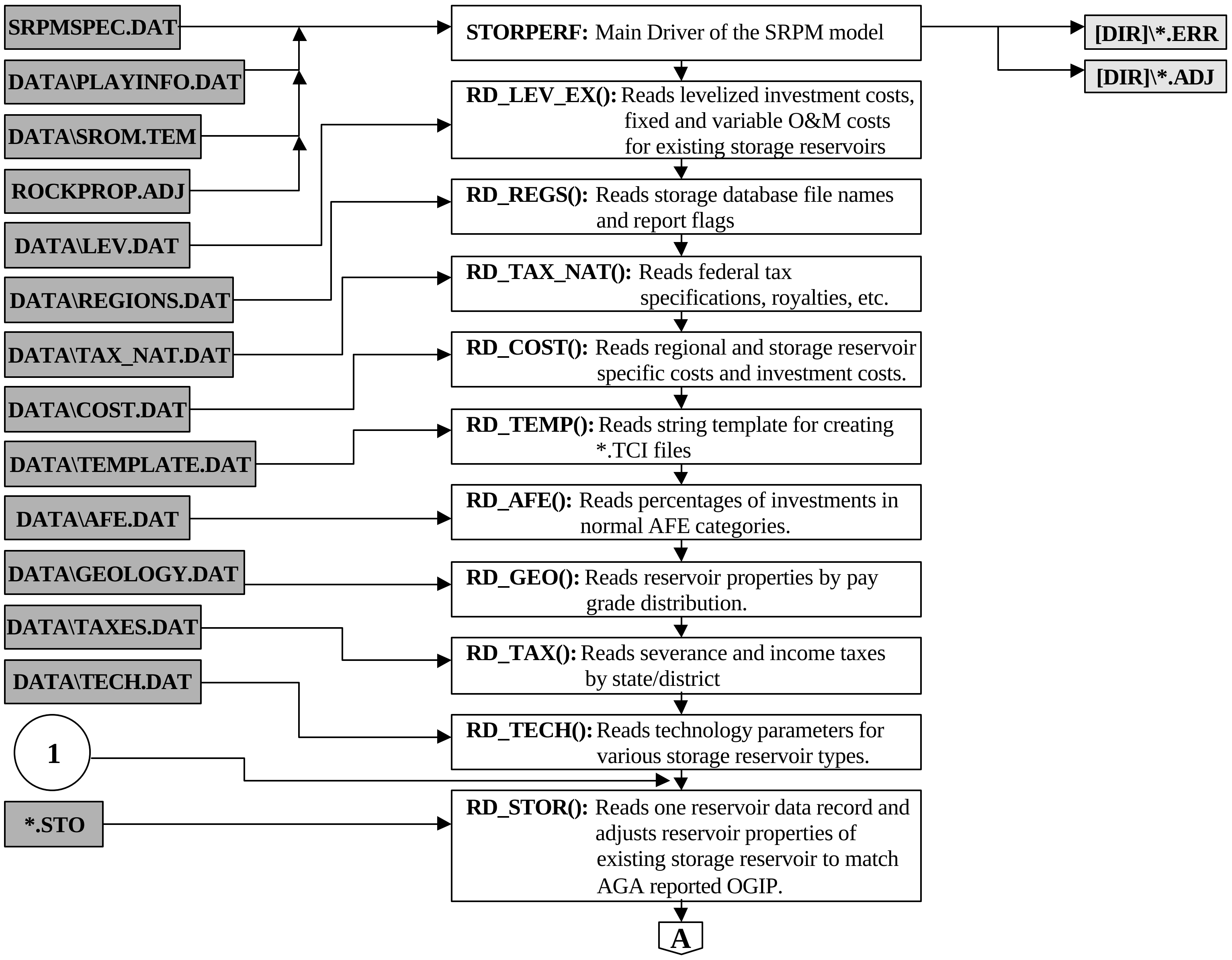
Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Assigned in	STORPERF.FOR	496
		Called by	RD_STOR.FOR	106, 107, 118, 119, 135, 138, 140, 168, 169, 170
		Called by	CONVERT.FOR	34
		Called by	STORPERF.FOR	478, 493, 506, 525
wlspadj	Adjusted well spacing (acres)	Declared in	RD_DATA.H	20, 37
		Called by	RD_STOR.FOR	167
		Called by	STORPERF.FOR	115
wrad_tech	Wellbore radius (ft)	Declared in	TECH.H	8, 19
		Assigned in	RD_TECH.FOR	62
		Called by	CONVERT.FOR	81
wspace	Well spacing (acres)	Declared in	TYPE4.H	1
		Assigned in	MK_TYPE.FOR	49
		Assigned in	INIT_WEL.FOR	43
		Assigned in	CONVERT.FOR	48, 65
		Called by	CONVLV.FOR	37, 70
		Called by	RATE1.FOR	101
		Called by	DATOUT.FOR	87
		Called by	CONVERT.FOR	51, 52, 54, 55, 56, 57, 64, 71, 72
		Called by	GET_TYPE.FOR	21
		Called by	SETUP.FOR	19
yr1	Number of years for tax credit on tangible investment (years)	Declared in	TAX_NAT.H	45, 56
		Assigned in	CASHFLOW.FOR	49
		Called by	CASHFLOW.FOR	225
		Called by	RDTAXNAT.FOR	129
yr2	Number of years for tax credit on intangible investment (years)	Declared in	TAX_NAT.H	47, 56
		Called by	CASHFLOW.FOR	110, 133, 242
		Called by	RDTAXNAT.FOR	134
yr3	Number of years for forgiveness for state taxes (years)	Declared in	TAX_REG.H	15, 17
		Assigned in	CASHFLOW.FOR	73
		Called by	CASHFLOW.FOR	173
		Called by	RDTAXNAT.FOR	147
yractv	4-Digit year activated for storage (years)	Declared in	RD_DATA.H	14, 35
		Assigned in	RD_STOR.FOR	62
		Called by	RD_STOR.FOR	11
		Called by	STORPERF.FOR	644
		Called by	WRT_DI.FOR	1, 4, 10
yrdisc	4-Digit discovery year (years)	Declared in	RD_DATA.H	14, 31
		Called by	RD_STOR.FOR	11
zary	Gas Z-factor array	Declared in	TYPE2.H	1
		Assigned in	REALGS.FOR	31
		Called by	CALCPQ.FOR	17, 19, 21, 55, 61, 85, 98, 99
		Called by	CONVLV.FOR	14, 23, 28

Variable Name	Description	Cross Reference		
		Process	File Name	Line Number(s)
		Called by	PWELL.FOR	73, 88, 108, 134, 159, 161
		Called by	RATE1.FOR	37, 39, 78
		Called by	SETUP.FOR	21
		Called by	ZEE.FOR	1, 2, 6, 12, 15

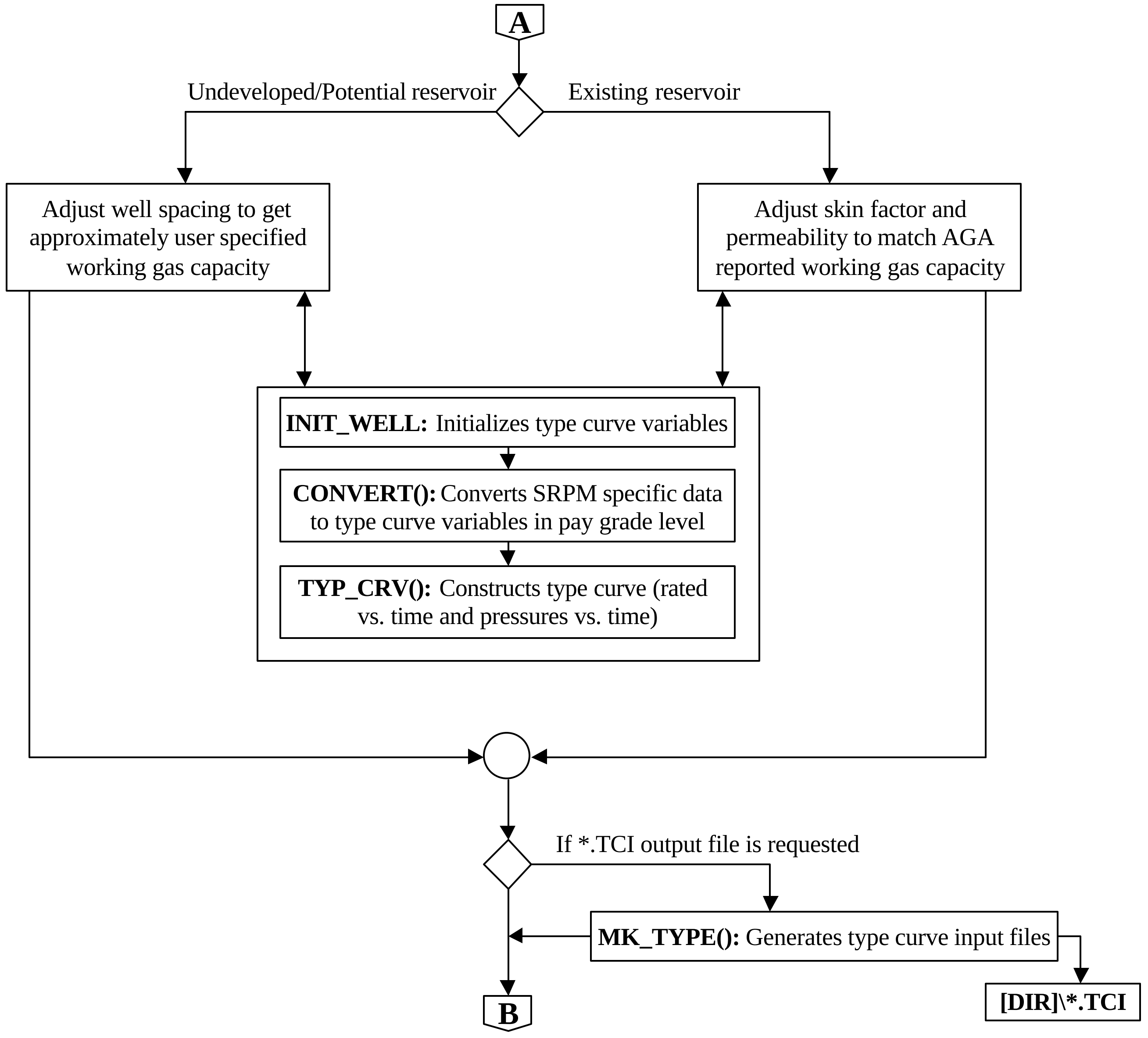
File Name	Location	Type*	Read/Write in	Description
*.ADJ	[MAIN] \[DIR]	O, Opt	STORPERF	One output file for each reservoir database in input file REGIONS.DAT that is generated to report adjusted reservoir properties.
*.ERR	[MAIN] \[DIR]	O	STORPERF	One output file for each reservoir database in input file REGIONS.DAT that is generated to report error/action messages.
*.PRD	[MAIN]\[DIR]	O, Opt	WRT_TCP()	One output file for each reservoir database in input file REGIONS.DAT that contains summary of rates and pressures.
*.PRO	[MAIN]\[DIR]	O, Opt	WRT_PRO()	One output file for each reservoir that contains cash flow pro-forma information.
*.SRO	[MAIN] \[DIR]	O	WRT_DI()	One output file for each reservoir database in input file REGIONS.DAT that contains reservoir performance data to be used in Demand and Integrating (D&I) Module.
*.STO	[MAIN]	I, Req	RD_STOR()	A set of storage reservoir database files that contain information on reservoir rock and fluid properties, wells data, etc. File names of the database files to be run through the SRPM has to be listed in input file REGIONS.DAT.
*.TCI	[MAIN]\[DIR]	O, Opt	MK_TYPE()	One output file for each reservoir that contains type curve input parameters.
*.TCO	[MAIN]\[DIR]	O, Opt	DATOUT()	One output file for each reservoir that contains type curve results.
AFE.DAT	[MAIN]\DATA	I, Req	RD_AFE()	Data of percentage of investment in normal AFE (authorization for expenditure) categories. Note that this information is not currently utilized in SRPM.
COST.DAT	[MAIN]\DATA	I, Req	RD_COST()	Data of regional and resource specific costs and investments.
DWLSPAC.DAT	[MAIN]\DATA	I, Req	RD_WSPAC()	Data of minimum well spacing for existing/potential storage reservoirs as a function of storage/demand region.
GEOLOGY.DAT	[MAIN]\DATA	I, Req	RD_GEO()	Data of reservoir property distributions by pay grade.
LEV.DAT	[MAIN]\DATA	I, Req	RD_LEV_EX()	Data of leveled cost, fixed and variable operating and maintenance costs for existing storage reservoirs based on operating company.
PLAYINFO.DAT	[MAIN]\DATA	I, Req	STORPERF	Data of play specific concentrations of gas impurities.
REGIONS.DAT	[MAIN]\DATA	I, Req	RD_REGS()	List of the *.STO files to be run through Storate Reservoir Performance Module (SRPM) and YES/NO flags for output files.
ROCKPROP.ADJ	[MAIN]	I, Opt	STORPERF	Data of adjusted permeability and skin factor. These secondary data are calculated based on matched between reported deliverability and computed deliverability.
SPRMSPEC.DAT	[MAIN]	I, Req	STORPERF	Data of SRPM run specifications.
SROM.TEM	[MAIN]\DATA	I, Opt	STORPERF	A template file that contains header lines for *.SRO output files.
TAX_NAT.DAT	[MAIN]\DATA	I, Req	RD_TAX_NAT()	Data of generic tax structure (capitalize versus expense switches)

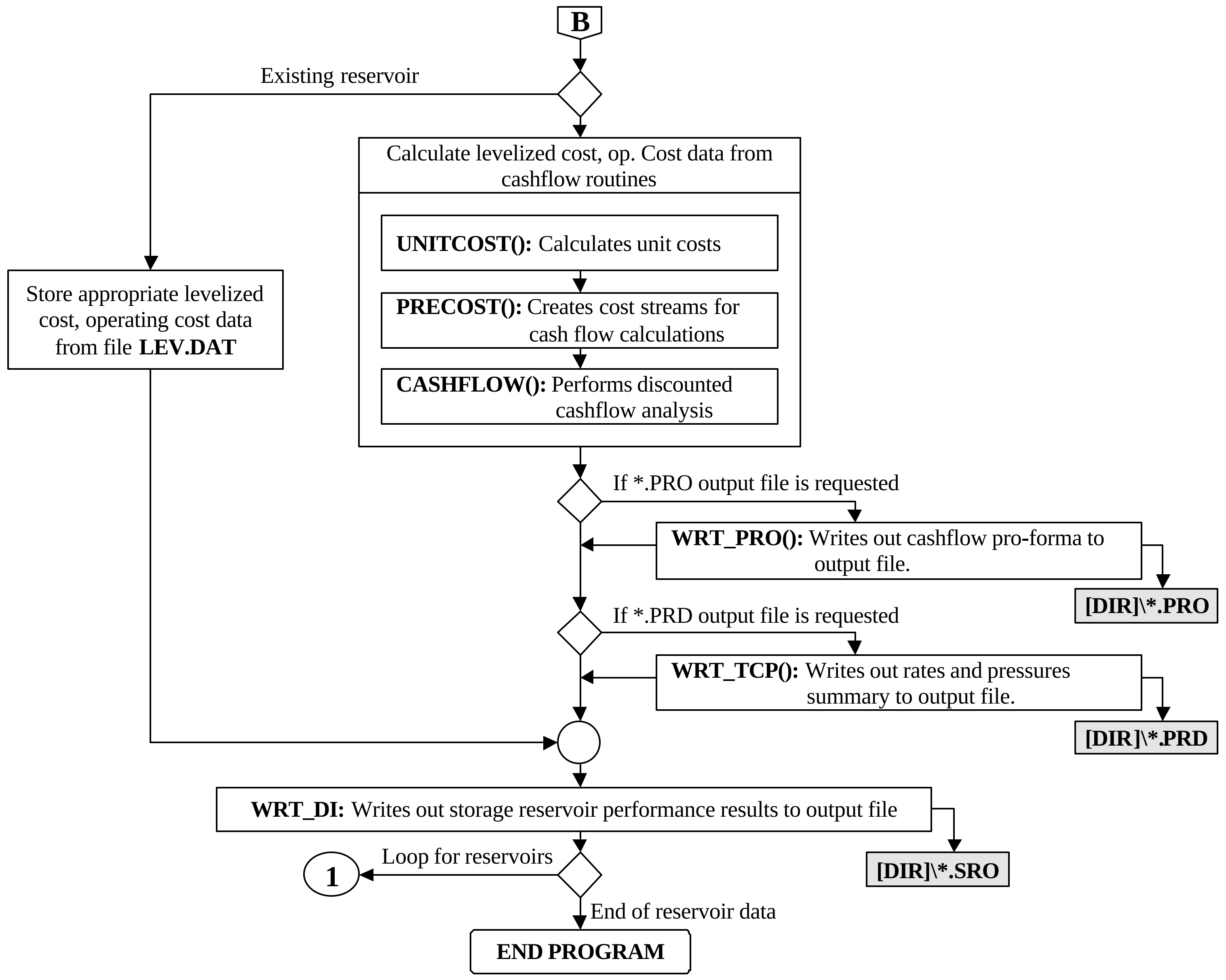
				assumptions.
TAXES.DAT	[MAIN]\DATA	I, Req	RD_TAX()	Data of state income taxes, oil and gas severance taxes, and ad-voleram taxes.
TECH.DAT	[MAIN]\DATA	I, Req	RD_TECH()	Data of technology specifications for various storage reservoir types.
TEMPLATE.DAT	[MAIN]\DATA	I, Req	RD_TEMP()	A template file used to generate type curve input parameters. It contains information on fluid and reservoir properties data, well data, field development, drive mechanism, and other type curve related data.

*I=Input file, O=Output file, Req=Required, Opt=Optional



SRPMSPEC.DAT
DATA\PLAYINFO.DAT
DATA\SROM.TEM
ROCKPROP.ADJ
DATA\LEV.DAT
DATA\REGIONS.DAT
DATA\TAX_NAT.DAT
DATA\COST.DAT
DATA\TEMPLATE.DAT
DATA\AFE.DAT
DATA\GEOLOGY.DAT
DATA\TAXES.DAT
DATA\TECH.DAT
1
*.STO
STORPERF: Main Driver of the SRPM model
[DIR]*.ERR
[DIR]*.ADJ
RD_LEV_EX(): Reads levelized investment costs, fixed and variable O&M costs for existing storage reservoirs
RD_REGS(): Reads storage database file names and report flags
RD_TAX_NAT(): Reads federal tax specifications, royalties, etc.
RD_COST(): Reads regional and storage reservoir specific costs and investment costs.
RD_TEMP(): Reads string template for creating *.TCI files
RD_AFE(): Reads percentages of investments in normal AFE categories.
RD_GEO(): Reads reservoir properties by pay grade distribution.
RD_TAX(): Reads severance and income taxes by state/district
RD_TECH(): Reads technology parameters for various storage reservoir types.
RD_STOR(): Reads one reservoir data record and adjusts reservoir properties of existing storage reservoir to match AGA reported OGIP.
A


A
Undeveloped/Potential reservoir
Existing reservoir
Adjust well spacing to get approximately user specified working gas capacity
Adjust skin factor and permeability to match AGA reported working gas capacity
INIT_WELL: Initializes type curve variables
CONVERT(): Converts SRPM specific data to type curve variables in pay grade level
TYP_CRV(): Constructs type curve (rated vs. time and pressures vs. time)
If *.TCI output file is requested
MK_TYPE(): Generates type curve input files
[DIR]*.TCI
B


B
Existing reservoir
Calculate levelized cost, op. Cost data from cashflow routines
UNITCOST(): Calculates unit costs
PRECOST(): Creates cost streams for cash flow calculations
CASHFLOW(): Performs discounted cashflow analysis
Store appropriate levelized cost, operating cost data from file LEV.DAT
If *.PRO output file is requested
WRT_PRO(): Writes out cashflow pro-forma to output file.
[DIR]*.PRO
If *.PRD output file is requested
WRT_TCP(): Writes out rates and pressures summary to output file.
[DIR]*.PRD
WRT_DI: Writes out storage reservoir performance results to output file
[DIR]*.SRO
1
Loop for reservoirs
End of reservoir data
END PROGRAM

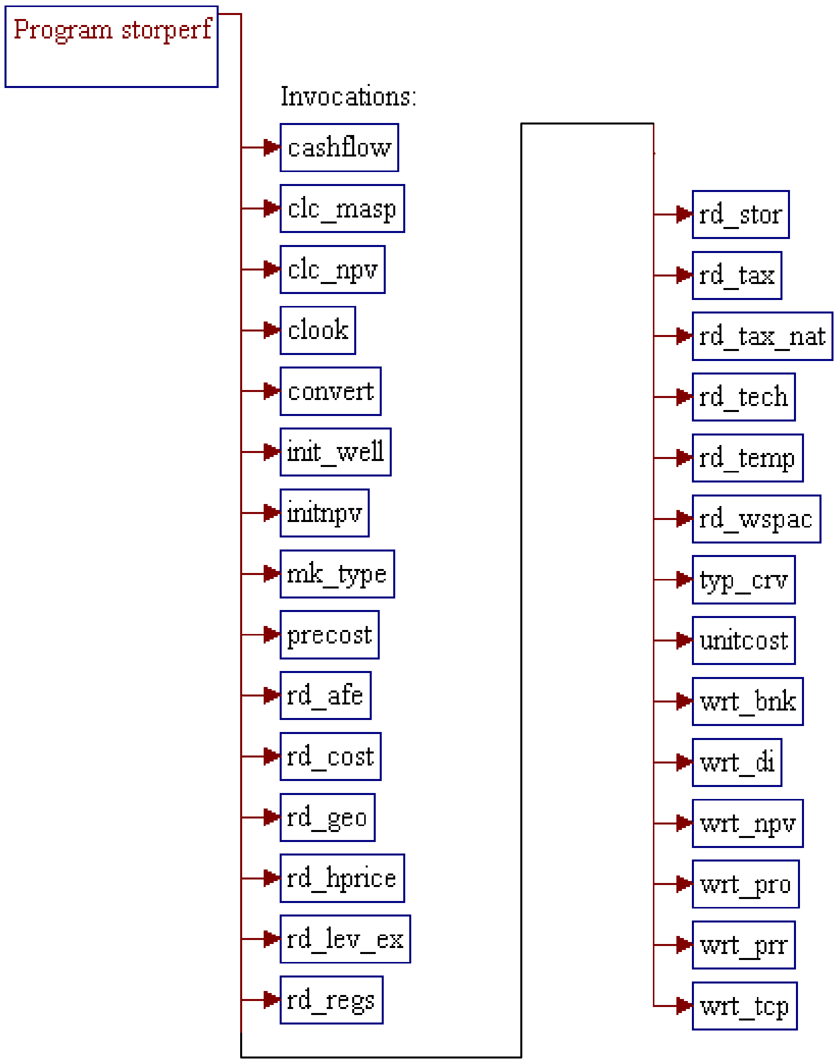
SUB-PROGRAM STORPERF()

MAIN THEME: This is the main program of the Storage Reservoir Performance Module (SRPM).

READS: SRPMSPEC.DAT
PLAYINFO.DAT
SROM.TEM
ROCKPROP.ADJ

CREATES: *.ADJ
*.ERR

ROUTINE INTERACTIONS:



Step 1: **Declarations and definitions.**

```
program storperf
```

Note: Include files, common block, and local variables.

```
include 'rd_data.h'
include 'dimen.h'
include 'global.h'
include 'cashflow.h'
include 'costing.h'
include 'cost.h'
include 'field.h'
include 'tax_nat.h'
include 'tax_reg.h'
include 'unitcost.h'
include 'gsamvar.h'
include 'welldata.h'
include 'type_out.h'
include 'tech.h'
include 'type1.h'
include 'type2.h'
include 'type3.h'
include 'type4.h'
include 'type5.h'
include 'type6.h'
include 'type7.h'
include 'type8.h'
include 'type9.h'
include 'type10.h'
include 'storlp.h'
common /stchg/iwin_yr
integer ireg,irec,iyrenv
integer ipa
integer storflag
real*4  denom
real*4 h2scon(1000),co2con(1000),n2con(1000),dummy
real*4 fom_tmp,vom_tmp,lcst_tmp
real*4 qg0(3),smin(3),smax(3)
character*4  cplay(1000)
character*30 fld3,coun3
character*190 char(20)
integer icode
character*255 dirname
```

Step 2: **Assign names of development types for reporting purposes. Note that the current SRPM model only considers the primary well option.**

```
casename(1)='Primary'
casename(2)='Refrac '
casename(3)='Infill '
```

Step 3: **Read data from *SRPMSPEC.DAT*:**
- *dirname* is name of output directory to store output files
- *iexruntyp* is an SRPM run type for existing storage reservoirs:
0 Create *ROCKPROP.ADJ*:

*** Perform permeability, skin, pay thickness adjustments to match reported AGA maximum deliverability and OGIP**
*** Store adjusted rock properties to *ROCKPROP.ADJ***
*** Run type curve module**
1 Read *ROCKPROP.ADJ*:
*** Read adjusted rock properties from *ROCKPROP.ADJ***
*** Run type curve module**
- *nyrset_storage* is number of years for potential storage mode run.
- *fracogip* is maximum working gas capacity (fraction of OGIP)

```
      open(unit=87,file='srpmspec.dat',status='old')
      read(87,*)
      read(87,'(a)') dirname
      read(87,*)
      read(87,*) iexruntyp
      read(87,*)
      read(87,*) nyrset_storage
      read(87,*)
      read(87,*) fracogip
      close(87)
      if (iexruntyp.lt.0.or.iexruntyp.gt.1) iexruntyp = 0
      if (fracogip.lt.0.0.or.fracogip.gt.1.0) fracogip = 0.8
```

Note: - *prod_period* is production/withdrawal period (days) and it is hardwired to 121 days.
- *deltat* is time step size (days) and it is hardwired to 1 day.
- *stor_fact* is cost of injection gas as a fraction of gas price and it is hardwired to 1.

```
      prod_period = 121.0
      deltat = 1.0
      stor_fact = 1.0
```

Note: Get location of last character *ild* (not a blank space) in the string *dirname.*

```
      ild = index(dirname,' ')
      i = makedirqq(dirname)
      dirname(ild:ild) = '\'
```

Note: Convert production period and time step size to years.

```
      prod_period = prod_period/365.0
      deltat = deltat/365.0
```

Step 4: **Subroutine *RD_WSPAC()* reads existing storage well spacing (used when data of number of wells is missing) from input file *DWLSPAC.DAT*.**

```
      open(unit=87,file='data\dwlspac.dat',status='old')
      call rd_wspac(87)
      close(87)
```

Step 5: **Subroutine *RD_LEV_EX()* reads levelized investment cost, fixed and variable O&M for existing storage reservoirs based on the operating company from input file *LEV.DAT*.**

```
      open(unit=87,file='data\lev.dat',status='old')
      call rd_lev_ex(87)
      close(87)
```

Step 6: **Subroutine *RD_REGS()* reads names of the storage database files from input file *REGIONS.DAT* to be used in the analysis.**

```
      open(unit=87,file='data\regions.dat',status='old')
      call rd_regs(87)
      close(87)
```

Step 7: **Subroutine *RD_TAX_NAT()* reads federal tax specifications, royalties etc. from input file *TAX_NAT.DAT*.**

```
      open(unit=87,file='data\tax_nat.dat',status='old')
      call rd_tax_nat(87)
      close(87)
```

Step 8: **Subroutine *RD_COST()* reads regional and storage reservoir specific costs and investment costs from input file *COST.DAT*.**

```
      open(unit=87,file='data\cost.dat',status='old')
      call rd_cost(87)
      close(87)
```

Step 9: **Subroutine *RD_TEMP()* reads string template from input file *TEMPLATE.DAT* to be displayed.**

```
open(unit=87,file='data\template.dat',status='old')
call rd_temp(87)
close(87)
```

Step 10: **Subroutine *RD_AFE()* reads percentages of investments in normal AFE categories from input file *AFE.DAT* (not currently used).**

```
open(unit=87,file='data\afe.dat',status='old')
call rd_afe(87)
close(87)
```

Step 11: **Subroutine *RD_GEO()* reads reservoir properties by pay-grade distribution from input file *GEOLOGY.DAT*. Note that current SRPM is set up to use only one pay grade per reservoir.**

```
open(unit=87,file='data\geology.dat',status='old')
call rd_geo(87)
close(87)
```

Step 12: **Subroutine *RD_TAX()* reads severance and income taxes by state/district from input file *TAXES.DAT*.**

```
open(unit=87,file='data\taxes.dat',status='old')
call rd_tax(87)
close(87)
```

Step 13: **Subroutine *RD_TECH()* reads technology parameters for various storage reservoir types from input file *TECH.DAT*.**

```
open(unit=87,file='data\tech.dat',status='old')
call rd_tech(87)
close(87)
```

Step 14: **Read play specific impurity level from input file *PLAYINFO.DAT*.**

```
open(unit=87,file='data\playinfo.dat',status='old')
read(87,*)
do ipa = 1,10000
  read(87,'(a4)',end=10) cplay(ipa)
  backspace(87)
```

```
        read(87,*) dummy,h2scon(ipa),co2con(ipa),n2con(ipa)
        h2scon(ipa)=h2scon(ipa)/100.
        co2con(ipa)=co2con(ipa)/100.
        n2con(ipa)=n2con(ipa)/100.
      end do
 10   ntotplay = ipa-1
      close(87)
```

Step 15: **Read header lines from template file *SROM.TEM* to be displayed in **.SRO* output files.**

```
      open(unit=87,file='data\srom.tem',status='old',err=20)
      do i = 1,11
        read(87,'(a190)',end=20) char(i)
      end do
 20   close(87)
```

Step 16: **If *iexruntyp=1*, read adjusted rock properties from file *ROCKPROP.ADJ*.**

Note: *gid* is 11-digit SRPM/GSAM ID
permadj is adjusted permeability (md)
poradj is adjusted porosity (fraction)
sgadj is adjusted gas saturation (fraction)
payadj is adjusted net pay thickness (ft)
skinadj is adjusted skin factor
acpradj is adjusted well drainage area (acres)
wlspadj is adjusted well spacing (acres)
peakrate is maximum deliverability (MMCF/D)

```
      if (iexruntyp.eq.1) then
        open(87,file='rockprop.adj')
        i = 1
        read(87,*)
 30     read(87,40,end=50) gid(i),permadj(i),poradj(i),sgadj(i),
     &    payadj(i),skinadj(i),acpradj(i),wlspadj(i),peakrate
 40     format(2x,a11,7(2x,f12.3))
        i = i+1
        goto 30
 50     ncis = i-1
        close(87)
      endif
```

Note: Stop the program if number of data is greater than 500.
Note: Size of arrays for adjusted rock properties is 500 (see *RD_DATA.H*).

```
      if (ncis.gt.500) then
        print*,'Number of reservoirs in file ROCKPROP.ADJ is ',
   &      'greater than 500 (out of range).'
        print*,'Change DIMENSION declaration in RD_DATA.H and ',
   &      'recompile the program.'
        stop
      endif
```

Step 17: **Loop on each file specified in input file *REGIONS.DAT*.**

Note: Reset counter of number of reservoir *irec* processed.

```
        irec = 1
```

Note: Initialize file (region) loop *ireg*.

```
        do 1000 ireg = 1,nreg
```

Step 18: **Get location of last character *ild* (not a blank space) in the prefix of reservoir database file name.**

```
          ilf = index(regnm(ireg),' ')-1
```

Step 19: **Open reservoir database file **.STO* to be read. After opening the file, read 4 header lines.**

```
          open(unit=10,file=regnm(ireg)(1:ilf)//'.sto',status='old')
          read(10,*)
          read(10,*)
          read(10,*)
          read(10,*)
```

Step 20: **Open primary SRPM output files (one file for each reservoir database).**

Note: File "*.ERR" (UNIT=501) is a list IDs of reservoirs:
- without pressure and depth data
- not storage reservoirs
- etc.

After opening the file, print a header line.

```
        open(unit=501,file=dirname(1:ild)//regnm(ireg)(1:ilf)//'.err',
     &    status='unknown',recl=500)
        write(501,*) 'File|GSAM ID|Action|Error message|'
```

Note: File **.ADJ* (UNIT=502) contains adjusted reservoir rock properties.
This file is created only if it is requested in *SRPMSPEC.DAT* (*iexruntyp=0*).
After opening the file, print a header line.

```
        if (iexruntyp.eq.0) then
          open(unit=502,
     &      file=dirname(1:ild)//regnm(ireg)(1:ilf)//'.adj',
     &      status='unknown')
          write(502,'(a2,a11,8(a2,a12))')
     &      '  ','GSAMID',
     &      '  ','PERM (md)',
     &      '  ','POR (frac)',
     &      '  ','SG (frac)',
     &      '  ','NET PAY (ft)',
     &      '  ','SKIN',
     &      '  ','AREA (acres)',
     &      '  ','SPC (acres)',
     &      '  ','PEAK (MMCFD)'
        endif
```

Note: File **.SRO* (UNIT=503) contains storage reservoir performance data to be used in Demand and Integrating (D&I) Module.
After opening the file, print header lines from template file *SROM.TEM.*

```
        open(unit=503,file=dirname(1:ild)//regnm(ireg)(1:ilf)//'.sro',
     @    status='unknown')
        do i = 1,11
          write(503,'(a190)') char(i)
        end do
```

Step 21: **Open other SRPM output files (one file for each reservoir database). Note that these files will not be used by D&I Module of GSAM.**

Note: File **.PRD* reports summary of rates, cumulative production, and pressures. This file is created only if it is requested in input file *REGIONS.DAT*. (One file for each reservoir database).

```
        if (l_prd) then
          open(unit=504,
     &      file=dirname(1:ild)//regnm(ireg)(1:ilf)//'.prd',
     &      recl=1000)
          endif
```

Step 22: **Set *itech=1* (primary wells only)**
Define line number 101 as the top of reservoir record loop.

```
          itech = 1
  101     This_Is_Top_Reservoir_Loop = 0
```

Step 23: **Subroutine *RD_STOR()* reads next record from reservoir database.**

Note: After reading the data, the following assignments are performed:
- If data of some parameters are missing, the values for these parameters are hardwired to specified constants.
- Values for some parameters are forced between specified ranges.
- If rock properties adjustment is requested in file *SRPMSPEC.DAT* for existing storage reservoir, values of drainage area (*acprod*), pay thickness (*netpay*), porosity (*por*), and well spacing (*wlspac*) are adjusted to get total OGIP equals to the one reported by AGA (*capacity*).
- Go to label 2000 if end of file is encountered.

```
          call rd_stor(10,*2000,icode,fld3,coun3)
```

Note: Store original value of skin factor specified in file *TECH.DAT*

```
          skin_org = fracsk_tech(itech,modulesrpm)
```

Step 24: **An error is found in current reservoir data:**
- Write reservoir GSAM ID and error message to output file **.ERR*.
- Skip reservoir performance calculation and continue to the next reservoir record.

```
          if (icode.eq.1) then
            write(501,*) regnm(ireg)(1:ilf),'|',gsamid(1:11),
     &      '|Skipped|Does not have pressure and depth data.|'
            print*,regnm(ireg)(1:ilf),',',gsamid(1:11),': ',
     &      'Does not have pressure and depth data. ',
     &      'Calculation is skipped.'
```

```
              goto 101
            else if (icode.eq.2) then
              write(501,*) regnm(ireg)(1:ilf),'|',gsamid(1:11),
     &          '|Skipped|Does not have maximum deliverability and ',
     &          'total working gas volume data.|'
              print*,regnm(ireg)(1:ilf),',',gsamid(1:11),': ',
     &          'Does not have maximum deliverability and ',
     &          'total working gas volume data. ',
     &          'Calculation is skipped.'
              goto 101
            else if (icode.eq.3) then
              write(501,*) regnm(ireg)(1:ilf),'|',gsamid(1:11),
     &          '|Skipped|Not a gas storage reservoir (module number ',
     &          'out of range).|'
              print*,regnm(ireg)(1:ilf),',',gsamid(1:11),': ',
     &          'Not a gas storage reservoir (module number ',
     &          'out of range).  Calculation is skipped.'
              goto 101
            else if (icode.eq.4) then
              write(501,*) regnm(ireg)(1:ilf),'|',gsamid(1:11),
     &          '|Skipped|Adjusted properties are not found or not ',
     &          'available in input file ROCKPROP.ADJ.|'
              print*,regnm(ireg)(1:ilf),',',gsamid(1:11),': ',
     &          'Adjusted properties are not found or not ',
     &          'available in input file ROCKPROP.ADJ.  ',
     &          'Calculation is skipped.'
              goto 101
            else if (icode.eq.5) then
              write(501,*) regnm(ireg)(1:ilf),'|',gsamid(1:11),
     &          '|Skipped|Well spacing data based on regional average ',
     &          'is required but not found in file DWLSPAC.DAT.|'
              print*,regnm(ireg)(1:ilf),',',gsamid(1:11),': ',
     &          'Well spacing data based on regional average ',
     &          'is required but not found in file DWLSPAC.DAT.  ',
     &          'Calculation is skipped.'
              goto 101
            else if (icode.eq.6) then
              write(501,*) regnm(ireg)(1:ilf),'|',gsamid(1:11),
     &          '|Skipped|Unable to match AGA reported storage capacity.|'
              print*,regnm(ireg)(1:ilf),',',gsamid(1:11),': ',
     &          'Unable to match AGA reported storage capacity.  ',
     &          'Calculation is skipped.'
              goto 101
            endif
```

Step 25: Check storage ID, and storage characteristics

Note:
- *module=7* is for depleted gas reservoirs
- *module=8* is for depleted water drive reservoirs
- *module=9* is for salt dome reservoirs

If the current reservoir is not a storage reservoir:
- Write reservoir GSAM ID and error message to output file **.ERR.*
- Skip reservoir performance calculation and continue to the next reservoir record.

```
            if ((module.ne.7).and.(module.ne.8).and.(module.ne.9)) then
              write(501,*) regnm(ireg)(1:ilf),'|',gsamid(1:11),
     &          '|Skipped|Not a gas storage reservoir.|'
              print*,regnm(ireg)(1:ilf),',',gsamid(1:11),': ',
     &          'Not a gas storage reservoir. Calculation is skipped.'
```

```
        goto 101
      endif
```

Step 26: **If actual concentration data is available in database, use the database values. Otherwise, take the USGS play average from input file *PLAYINFO.DAT*. If no match is found in the USGS data:**
- Write reservoir GSAM ID and error message to output file **.ERR*.
- Skip reservoir performance calculation and continue to the next reservoir record.

```
      if ((co2+n2+h2s).le.0.0) then
        call clook(gsamid(5:8),cplay,ntotplay,ipa)
        if (ipa.eq.0) then
          write(501,*) regnm(ireg)(1:ilf),'|',gsamid(1:11),
   &        '|Defaulted|Concentration (H2S+CO2+N2)<0 and no ',
   &        'match is found in file PLAYINFO.DAT.|'
          h2s = 0.0
          co2 = 0.0
          n2 = 0.0
        else
          h2s = h2scon(ipa)
          co2 = co2con(ipa)
          n2 = n2con(ipa)
        endif
      endif
```

Note: Print some information about the reservoir being analyzed to the console.

```
      write(6,'(t2,a,a,a,a,a,i4)')
   &   'File: ',regnm(ireg),' ID: ',gsamid,' Total processed:',irec
```

Step 27: **Increment counter for number of reservoirs processed (total from start).**

```
      irec = irec+1
```

Step 28: **Open diagnostic files requested in input file *REGIONS.DAT*. (One file for each reservoir).**

```
      if (l_tco) then
        open(unit=505,file=dirname(1:ild)//gsamid(4:11)//'.tco')
      endif
      if (l_tci) then
        open(unit=506,file=dirname(1:ild)//gsamid(4:11)//'.tci')
```

```
          endif
          if (l_pro) then
            open(unit=507,file=dirname(1:ild)//gsamid(4:11)//'.pro')
          endif
```

Step 29: **CASE #1: Existing storage reservoirs (*statin=0*) and non-technology run (*iexruntyp=0*):**

Note: * Perform permeability and skin adjustments to match reported AGA working gas.
* Store adjusted rock properties to *ROCKPROP.ADJ*.
* Run type curve module.

```
          if (statin.eq.0.and.iexruntyp.eq.0) then
```

Step 30: **First, iterate on skin factor *fracsk_tech* using a maximum of 50 bisection iterations.**

Note: *modulesrpm* is the original module number (7,8, or 9).
Subroutine *INIT_WELL* initializes variables.
Subroutine *CONVERT()* converts SRPM specific data to type curve variable names in pay grade level.
Subroutine *TYP_CRV()* constructs type curve (rate vs. time).
qg is gas rate (MCF/D/Well).
type_well is number of wells.
totgasprod is calculated total gas produced (MMCF/D).
maxdeliv is AGA reported maximum deliverability (MMCF/D).
mwgc is calculated maximum total working gas capacity (MMCF)
totwork is AGA reported maximum total working gas capacity (MMCF)

```
            write(6,'(a,a,f7.0,a)') '      SRPM ITERATES ON SKIN FACTOR ',
     &        'TO MATCH WORKING GAS OF :',totwork,' MMCF'
            print*,'      Iter#   Skin Factor   Working Gas (MMCF)',
     &        '   Error (%)'
```

Note: Get range of skin factor *smin* and *smax* for working gas iteration.

```
            do i = 1,3
              smin(i) = 0.0
              smax(i) = 0.0
            enddo
            tgmin = 0.0
            tgmax = 0.0
            call init_well
```

```
           call convert(itech,pg1fact,pg3fact)
           ratmax = 1.0
           call typ_crv(gsamid(4:11),1,1,1,l_tco,maxtim)
           totgasprod = 0.0
           mwgc = 0.0
           do i = 1,3
             j = 1
             totgasprod = totgasprod+qg(i,j,1)*type_well(j,i)/1000.0
             mwgc = mwgc+cumgas(i,j,maxtim)*type_well(j,i)/1000.0
           enddo
           funcdir = mwgc-totwork
           if (funcdir.gt.0.0) then
             do i = 1,3
               smin(i) = skin(i,1,1)
             enddo
             sbase = smin(2)
             tgmin = mwgc
             funcdir = 1.0
           else
             do i = 1,3
               smax(i) = skin(i,1,1)
             enddo
             sbase = smax(2)
             tgmax = mwgc
             funcdir = -1.0
           endif
           iter = 0
   105     iter = iter+1
           call init_well
           call convert(itech,pg1fact,pg3fact)
           ratmax = 1.0
           sbase = sbase+funcdir*2.0
           if (sbase.lt.-15.0) sbase = -15.0
           if (sbase.gt.15.0) sbase = 15.0
           do i = 1,3
             skin(i,1,1) = sbase
           enddo
           call typ_crv(gsamid(4:11),1,1,1,l_tco,maxtim)
           totgasprod = 0.0
           mwgc = 0.0
           do i = 1,3
             j = 1
             totgasprod = totgasprod+qg(i,j,1)*type_well(j,i)/1000.0
             mwgc = mwgc+cumgas(i,j,maxtim)*type_well(j,i)/1000.0
           enddo
           funcnew = mwgc-totwork
           if ((abs(funcnew)/totwork).lt.0.03) then
             fracsk_tech(itech,modulesrpm) = sbase
             goto 150
           endif
           if (funcnew.gt.0.0) then
             do i = 1,3
               smin(i) = sbase
             enddo
             tgmin = mwgc
           else
             do i = 1,3
               smax(i) = sbase
             enddo
             tgmax = mwgc
           endif
           write(6,'(6x,i5,3x,f11.1,3x,f18.1,3x,f9.1)') iter,sbase,
     &       mwgc,abs(funcnew)/mwgc*100.0
           if ((funcnew*funcdir).le.0.0) goto 110
           if (sbase.le.-15.0.or.sbase.ge.15.0) goto 115
           if (iter.lt.50) goto 105
           fracsk_tech(itech,modulesrpm) = sbase
           goto 150
```

Note: Iterate on skin factor to match working gas *totwork*.

```
110       iter = iter+1
          call init_well
          call convert(itech,pg1fact,pg3fact)
          ratmax = 1.0
          do i = 1,3
            skin(i,1,1) = (maxdeliv-tgmin)/(tgmax-tgmin)*
     &        (smax(i)-smin(i))+smin(i)
          enddo
          sbase = skin(2,1,1)
          call typ_crv(gsamid(4:11),1,1,1,l_tco,maxtim)
          totgasprod = 0.0
          mwgc = 0.0
          do i = 1,3
            j = 1
            totgasprod = totgasprod+qg(i,j,1)*type_well(j,i)/1000.0
            mwgc = mwgc+cumgas(i,j,maxtim)*type_well(j,i)/1000.0
          enddo
          funcnew = mwgc-totwork
          if (funcnew.gt.0.0) then
            do i = 1,3
              smin(i) = sbase
            enddo
            tgmin = mwgc
          else
            do i = 1,3
              smax(i) = sbase
            enddo
            tgmax = mwgc
          endif
          write(6,'(6x,i5,3x,f11.1,3x,f18.1,3x,f9.1)') iter,sbase,
     &      mwgc,abs(funcnew)/mwgc*100.0
          if ((abs(funcnew)/maxdeliv).lt.0.03) then
            fracsk_tech(itech,modulesrpm) = sbase
            goto 150
          endif
          if (iter.lt.50) goto 110
```

Step 31: **Iterate on permeability *perhor* to match *totwork* if skin is out of range. Use either *skinmin* or *skinmax* as skin factor depending on which one gives the smallest absolute skin function.**

Note: *perhor* is horizontal permeability (md).
pervrt is vertical permeability (md) (30% of *perhor*).
permtx is matrix permeability (md) (10% of *perhor*).

Set skin factor and base permeability function.

```
115       continue
          write(6,'(a,a,f7.0,a)') '      SRPM ITERATES ON PERMEABILI',
     &      'TY TO MATCH WORKING GAS OF :',totwork,' MMCF'
          print*,'      Iter#    Perm. (md)   Working Gas (MMCF)',
     &      '   Error (%)'
          fracsk_tech(itech,modulesrpm) = sbase
          funcbase = funcnew
          iter = 0
```

Note: Get range of permeability for bisection iteration: *perbase* and *perlimit*.

```
          if (funcbase.lt.0.0) then
            facimprv = 1.5
            perlimit = 10000.0
          else
            facimprv = 0.5
            perlimit = 0.001
          endif
120       iter = iter+1
          perbase = perhor
          perhor = facimprv*perhor
          pervrt = 0.3*perhor
          permtx = 0.1*perhor
          call init_well
          call convert(itech,pg1fact,pg3fact)
          ratmax = 1.0
          do i = 1,3
            skin(i,1,1) = sbase
          enddo
          call typ_crv(gsamid(4:11),1,1,1,l_tco,maxtim)
          totgasprod = 0.0
          mwgc = 0.0
          do i = 1,3
            j = 1
            totgasprod = totgasprod+qg(i,j,1)*type_well(j,i)/1000.0
            mwgc = mwgc+cumgas(i,j,maxtim)*type_well(j,i)/1000.0
          enddo
          funcnew = mwgc-totwork
          write(6,'(6x,i5,3x,f11.1,3x,f18.1,3x,f9.1)') iter,perhor,
     &      mwgc,abs(funcnew)/mwgc*100.0
          if ((funcnew*funcbase).gt.0.0) then
            icode = 0
            if (facimpr.lt.1.0.and.perhor.lt.perlimit) icode = 1
            if (facimpr.gt.1.0.and.perhor.gt.perlimit) icode = 1
            if (icode.eq.1) then
              write(501,*) regnm(ireg)(1:ilf),'|',gsamid(1:11),
     &          '|Skipped|Unable to match AGA working gas.|'
              print*,regnm(ireg)(1:ilf),',',gsamid(1:11),': ',
     &          'Unable to match AGA working gas. ',
     &          'Calculation is skipped.'
              goto 101
            endif
            funcbase = funcnew
            goto 120
          endif
          funclimit = funcnew
          perlimit = perhor
```

Note: At this point, the root of permeability is expected between *perbase* and *perlimit* because *funcbase* and *funclimit* have different signs. Find the root using bisection iteration.

```
130       iter = iter+1
          pernew = 0.5*(perbase+perlimit)
          perhor = pernew
          pervrt = 0.3*perhor
          permtx = 0.1*perhor
```

```
      call init_well
      call convert(itech,pg1fact,pg3fact)
      ratmax = 1.0
      do i = 1,3
        skin(i,1,1) = sbase
      enddo
      call typ_crv(gsamid(4:11),1,1,1,l_tco,maxtim)
      totgasprod = 0.0
      mwgc = 0.0
      do i = 1,3
        j = 1
        totgasprod = totgasprod+qg(i,j,1)*type_well(j,i)/1000.0
        mwgc = mwgc+cumgas(i,j,maxtim)*type_well(j,i)/1000.0
      enddo
      funcnew = mwgc-totwork
      write(6,'(6x,i5,3x,f11.1,3x,f18.1,3x,f9.1)') iter,perhor,
   &    mwgc,abs(funcnew)/mwgc*100.0
      if ((abs(funcnew)/mwgc).lt.0.03) then
        fracsk_tech(itech,modulesrpm) = sbase
        goto 150
      endif
      if ((funcnew*funcbase).gt.0.0) then
        perbase = pernew
      else
        perlimit = pernew
      endif
      if (iter.lt.50) goto 130
150   continue
```

Note: Calculate peak rate *peakrate* (MMCF/D) using quadratic fit extrapolation of flow rates at time steps 1,2, and 3, to get flow rate at time 0.

```
      do i = 1,3
        j = 1
        x1 = time(1)
        x2 = time(2)
        x3 = time(3)
        y1 = qg(i,j,1)
        y2 = qg(i,j,2)
        y3 = qg(i,j,3)
        aa = (y1-2.0*y2+y3)/(2.0*(x1*x1-x2*x2)-(x1*x1-X3*x3))
        bb = (y1-y2-(x1*x1-x2*x2)*aa)/(x1-x2)
        cc = y1-x1*x1*aa-x1*bb
        qg0(i) = cc*type_well(j,i)/1000.0
      enddo
      peakrate = max(qg0(1),qg0(2),qg0(3))
```

Note: Store adjusted properties to output file *ROCKPROP.ADJ*

```
      write(502,'(2x,a11,8(2x,f12.3))')
   &    gsamid,
   &    perhor,
   &    portot,
   &    gassat,
   &    netpay,
   &    fracsk_tech(itech,modulesrpm),
   &    acprod,
   &    wlspac,
   &    peakrate
```

Note: Recall original skin factor specified in input file *TECH.DAT*

```
           fracsk_tech(itech,modulesrpm) = skin_org
```

Step 32: CASE #2: Existing storage reservoirs (*statin=0*) and technology run (*iexruntyp=1*)
*** Adjusted rock properties have been read from *ROCKPROP.ADJ*.**
*** Use absolute open flow as maximum total allowable gas rate *ratmax* (see subroutine *CONVERT()*).**
*** Run type curve module.**

```
        else if (statin.eq.0.and.iexruntyp.eq.1) then
          call init_well
          call convert(itech,pg1fact,pg3fact)
          call typ_crv(gsamid(4:11),1,1,1,l_tco,maxtim)
```

Step 33: CASE #3: Undeveloped/potential storage reservoirs (*statin=1*):
*** Find well spacing *wlspac* within 20 to 640 acres that can give maximum working gas capacity (if possible about 80%-90% of OGIP).**
*** Perform economic calculation**

Note: *mwgc* is maximum total working gas capacity (MMCF)
cumgas is cumulative gas production (MCF/Well)
togip is total original gas in place (MMCF)
IMPORTANT:
At the end of *wlspac* adjustment, calculated number of wells *type_well* will be different with database number of wells *dbwells*. If *type_well* is greater than *dbwells*, more wells need to be drilled. This will affect the costing calculation (see subroutine *PRECOST*).

```
        else if (statin.eq.1) then
          write(6,'(2a,i3,a,f9.0,a)') '      SRPM ITERATES ON WELL ',
     &      'SPACING TO GET ',nint(fracogip*100),'% OF :',togip,
     &      ' MMCF (OGIP)'
          print*,'      Iter#   Well Spacing (acres)   Working Gas',
     &      ' (MMCF)   %OGIP'
          deltawspc = 20.0
          wlspacnew = wlspac
          iter = 0
 160      iter = iter+1
          wlspac = wlspacnew
          call init_well
          call convert(itech,pg1fact,pg3fact)
          ratmax = 1.0
          call typ_crv(gsamid(4:11),1,1,1,l_tco,maxtim)
          mwgc = 0.0
          do i = 1,3
            j = 1
            mwgc = mwgc+cumgas(i,j,maxtim)*type_well(j,i)/1000.0
          enddo
          write(6,'(6x,i5,3x,f20.1,3x,f18.1,3x,f5.1)') iter,wlspac,
     &      mwgc,(mwgc/togip)*100.0
          if (wlspacnew.le.20.0.or.wlspacnew.ge.640) then
            write(501,'(4a,i4,a,i3,a)') regnm(ireg)(1:ilf),'|',
     &        gsamid(1:11),'|Use ',nint(wlspacnew),
     &        ' (acres)|Unable to achieve ',nint(fracogip*100),
     &        '% of OGIP.|'
            print*,regnm(ireg)(1:ilf),',',gsamid(1:11),': ',
     &        'Unable to achieve ',nint(fracogip*100),'% of OGIP.'
            goto 170
          endif
          dirfrac = mwgc/togip-fracogip
          if (iter.eq.1) dirfracold = dirfrac
          if (abs(dirfrac)/fracogip.gt.0.03) then
            if ((dirfrac*dirfracold).lt.0.0) deltawspc = 0.5*deltawspc
            if (deltawspc.lt.3.0) goto 170
            dirfracold = dirfrac
            facimpr = deltawspc
            if (dirfrac.le.0.0) facimpr = -deltawspc
            wlspacnew = wlspac+facimpr
            if (wlspacnew.lt.20.0) wlspacnew = 20.0
            if (wlspacnew.gt.640.0) wlspacnew = 640.0
            goto 160
          endif
        endif
170     continue
```

Step 34: **Subroutine *MK_TYPE()* generates type curve input file **.TCI***

```
        if (l_tci) call mk_type(506,itech,maxtim)
```

Step 35: ***type_work* is working gas capacity in a pay grade (MMCF).**
***type_base* is base gas capacity in a pay grade (MMCF).**
***type_well* is number of wells in a pay grade.**
***mwgc* is maximum total working gas capacity (MMCF).**
***mbgc* is minimum total base gas capacity (MMCF).**
***qrate* is total production rate in a pay grade (MMCF/D).**
***cumpay* is cumulative production in a pay grade (MMCF)**
***comp_fs* is compressor fuel and gas shrinkage factor (fraction)**
***comp_vc* is data for *comp_fs* from input file *COST.DAT* (fraction)**
***fus* is percentage of compressor fuel and gas shrinkage (%)**
***qinjrate* is injection rate in pay grade #2 (MMCF/D).**
***gasprod* is gas production (BCF/Yr)**
***gasinje* is gas injection (BCF/Yr)**

```
        mwgc = 0.0
        mbgc = 0.0
        do i = 1,3
          j = 1
          type_work(j,i) = cumgas(i,j,maxtim)*type_well(j,i)/1000.0
          type_base(j,i) = ogip1(i)*type_well(j,i)/1000.0-
     &      type_work(j,i)
          mwgc = mwgc+type_work(j,i)
          mbgc = mbgc+type_base(j,i)
        enddo
        if (mbgc.le.0.0) then
          mwgc = togip*0.90
          mbgc = togip*0.10
        endif

        do istep = 1,maxtim
          qrate(istep) = 0.0
          cumpay(istep) = 0.0
          do i = 1,3
            j = 1
            qrate(istep) = qrate(istep)+
     &        qg(i,j,istep)*type_well(j,i)/1000.0
            cumpay(istep) = cumpay(istep)+
     &        cumgas(i,j,istep)*type_well(j,i)/1000.0
          enddo
        enddo
        comp_fs = comp_vc(1,gsamsr)
        if (comp_fs.le.0.) comp_fs = comp_vc(1,qreg+1)
        fus = comp_fs*100.0
        qinjrate = type_work(1,2)*(1.0+comp_fs)/(365.0-prod_period)
        do iyr = 1,nyrset_storage
          gasprod(iyr) = mwgc/1000.0
          gasinje(iyr) = qinjrate*(365.0-prod_period)/1000.0
        enddo
```

Step 36: **Production/injection profile option #1:**

Season	Description	Rate (MMCF/D)	Period
1	production	*qrate_op1_1*	0 - 5
2	production	*qrate_op1_2*	5 - 31
3	production	*qrate_op1_3*	31 - 121
4	injection	*qinjrate*	121 - 365

Production/injection profile option #2:

Season	Description	Rate (MMCF/D)	Period
1	production	*qrate_op2_1*	0 - 31
2	production	*qrate_op2_2*	31 - 121
3	injection	*qinjrate*	121 - 365

Production/injection profile option #3:

Season	Description	Rate (MMCF/D)	Period
1	production	*qrate_op3_1*	0 - 121
3	injection	*qinjrate*	121 - 365

Note: The production/injection profile is currently based on 1 day time step.

```
qrate_op1_1 = cumpay(5)/5.0
qrate_op1_2 = (cumpay(31)-cumpay(5))/(31.0-5.0)
qrate_op1_3 = (cumpay(121)-cumpay(31))/(121.0-31.0)
qrate_op2_1 = cumpay(31)/31.0
qrate_op2_2 = (cumpay(121)-cumpay(31))/(121.0-31.0)
qrate_op3_1 = cumpay(121)/121.0
```

Step 37: **Set maximum extraction capacity (MECP) (%) of each season in each option:**
- *mecp_op1_1* is MECP of option 1 in season 1.
- *mecp_op1_2* is MECP of option 1 in season 2.
- *mecp_op1_3* is MECP of option 1 in season 3.
- *mecp_op2_1* is MECP of option 2 in season 1.
- *mecp_op2_2* is MECP of option 2 in season 2.
- *mecp_op3_1* is MECP of option 3 in season 1.
Set maximum injection capacity *micp* (%) of each season in each option.

```
if (mwgc.gt.0.0) then
  mecp_op1_1 = qrate_op1_1*100.0/mwgc
  mecp_op1_2 = qrate_op1_2*100.0/mwgc
  mecp_op1_3 = qrate_op1_3*100.0/mwgc
  mecp_op2_1 = qrate_op2_1*100.0/mwgc
  mecp_op2_2 = qrate_op2_2*100.0/mwgc
  mecp_op3_1 = qrate_op3_1*100.0/mwgc
  micp = qinjrate*100/mwgc
else
  mecp_op1_1 = 0.0
  mecp_op1_2 = 0.0
  mecp_op1_3 = 0.0
  mecp_op2_1 = 0.0
  mecp_op2_2 = 0.0
  mecp_op3_1 = 0.0
  micp = 0.0
endif
```

Step 38: **For undeveloped/potential storage reservoir (*statin=1*), perform cash flow analysis to calculate:**
- *fom* fixed O&M cost
- *vom* variable O&M cost
- *lcst1* levelized investment cost
- tt NPV of total working gas that will be handled for next nyrset_storage years for potential storage reservoir (calculated in *CASHFLOW.FOR*)

```
if (statin.eq.1) then
```

Note: Calculate peak rate *peakrate* (MCF/D/Well) to be used to design compressors:
- *peakrate1* is peak production rate which is based on linear extrapolation of production rates *qg* in the first and second time steps to get production rate at time zero *qg0*
- *peakrate2* is peak injection rate.

```
          do i = 1,3
            j = 1
            x1 = time(1)
            x2 = time(2)
            x3 = time(3)
            y1 = qg(i,j,1)
            y2 = qg(i,j,2)
            y3 = qg(i,j,3)
            aa = (y1-2.0*y2+y3)/(2.0*(x1*x1-x2*x2)-(x1*x1-X3*x3))
            bb = (y1-y2-(x1*x1-x2*x2)*aa)/(x1-x2)
            cc = y1-x1*x1*aa-x1*bb
            qg0(i) = cc
          enddo
          peakrate1 = max(qg0(1),qg0(2),qg0(3))
          do i = 1,3
            j = 1
            qg0(i) = cumgas(i,j,maxtim)*(1.0+comp_fs)/
     &        (365.0-prod_period)
          enddo
          peakrate2 = max(qg0(1),qg0(2),qg0(3))
          peakrate = max(peakrate1,peakrate2)
```

Note: Initialize costs.

```
          fom = 0.0
          vom = 0.0
          lcst1 = 0.0
          tt = 0.0
```

Note: *icase=1* (for primary wells only).
itech=1 (for current technology only).
storflag=1 (for primary wells and current technology only).
modulesrpm is the original reservoir module number.

```
          icase = 1
          itech = 1
          storflag = 1
          module = modulesrpm
          do ipay = 1,3
```

Note: Subroutine *UNITCOST()* calculates unit costs expressed either as MM$/Well or $/MCF:
- *nwell* is calculated number of wells (primary wells).

- *gprice(1,...)* is gas sales price ($2.00/MCF).
- *gprice(2,...)* is gas fuel usage price ($2.00/MCF)..

```
      nwell = type_well(1,ipay)
      do iyr = 1,nyrset_storage
        gprice(1,iyr)= 2.0
        gprice(2,iyr)= 2.0
      enddo
      call unitcost(itech)
```

Note: Subroutine *PRECOST()* utilizes the calculated unit cost data to create cost streams to be fed to cash flow routine *CASHFLOW().*

```
      call precost(itech,icase,iyrenv,storflag,fom_tmp,
     &  vom_tmp,ipay,aaa)
```

Note: Calculate *fom* and *vom*

```
      fom = fom+fom_tmp
      vom = vom+vom_tmp
```

Note: Subroutine *CASHFLOW()* performs discounted cash flow analysis:
- *lcst_tmp* is levelized investment cost for current pay grade.
- *maxcf* is a flag for environmental run (not durrently used).

```
      lcst_tmp = 0.0
      maxcf = 0
      call cashflow(itech,maxcf,storflag,lcst_tmp,denom)
```

Note: Calculate *lcst1.*

```
      lcst1 = lcst1+lcst_tmp
```

Note: Calculate *tt*

```
      tt = tt + denom
```

Step 39: **Subroutine *WRT_PRO()* is invoked to write out cash flow pro-forma to output file *.*PRO*. This file is created only if it is**

requested in input file *REGIONS.DAT*. Currently, only pay grade #2 is reported.

```
636                 if (l_pro.and.ipay.eq.2)
637          &        call wrt_pro(507,itech,icase,ipay)
```

Step 40: **Subroutine *WRT_TCP()* is invoked to write out rates, cumulative production, and pressures to output file **.PRD*. This file is created only if it is requested in input file *REGIONS.DAT*. Currently, only pay grade #2 is reported.**

Note: - *icounter* is correction year (not currently used).

```
638                 if (l_prd.and.ipay.eq.2) then
639                   icounter = 0
640                   call wrt_tcp(504,maxtim)
641                 endif
642               enddo
643             endif
```

Step 41: **Subroutine *WRT_DI()* is invoked to write out storage reservoir performance data to output file **.SRO* to be used in Demand and Integrating (D&I) Module.**

```
644             call wrt_di(yractv,tt,icomp)
```

Step 42: **Close **.TCO* (file unit #505), **.TCI* (file unit #506), and **.PRO* (file unit #507) files.**

```
645             if (l_tco) close(505)
646             if (l_tci) close(506)
647             if (l_pro) close(507)
```

Step 43: **Go back to the top of reservoir record loop (line #101) to process the next reservoir.**

```
648             goto 101
```

Step 44: **Close reservoir database file **.STO* (file unit #10) and **.PRD* file (file unit #504).**

```
2000    close(10)
        if (l_prd) close(504)
```

Step 45: **Continue to the next region file specified in input file *REGIONS.DAT*.**

```
1000  continue
```

Step 46: **Close the **.PRD* file (unit file #504) of the last region file.**

```
      close(504)
      stop
      end
```

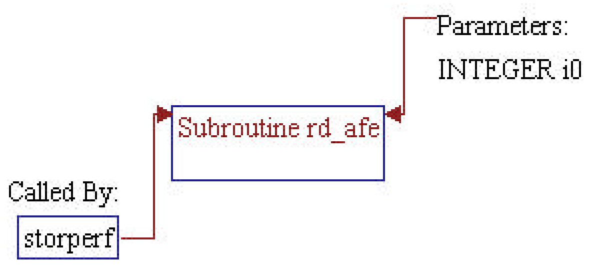
SUB-PROGRAM RD_AFE()

MAIN THEME: Reads input file AFE.DAT. Note that the current version of SRPM model does not yet implement the information given in input file AFE.DAT.

READS: AFE.DAT

CREATES: None

ROUTINE INTERACTIONS:



Step 1: Subroutine declarations and definitions.

Note: Parameter of the subroutine:

- *io* Unit number of input file AFE.DAT

```
      subroutine rd_afe(i0)
```

Note: Include files and local variable.

```
      include 'dimen.h'
      include 'unitcost.h'
      integer i0
```

Step 2: Read AFE data.

```
      read(i0,*)
      read(i0,*)
      read(i0,*)
      nafe=1
  100 read(i0,'(t1,a,t35,f6.0)',end=200) afename(nafe),afe(nafe)
      afe(nafe)=afe(nafe)/100
      nafe=nafe+1
      goto 100
  200 continue
      nafe=nafe-1
      return
      end
```

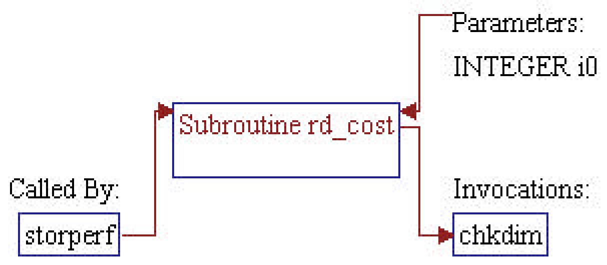
SUB-PROGRAM RD_COST()

MAIN THEME: Reads input file COST.DAT. In SRPM, the cost data is used to calculate levelized investment costs and fixed and variable operating and maintenance costs of the potential (undeveloped) storage reservoirs.

READS: COST.DAT

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameter of the subroutine:

- *io* Unit number of input file COST.DAT

```
      subroutine rd_cost(i0)
```

Note: Include files and local variables.

```
      include 'dimen.h'
      include 'cost.h'
      include 'field.h'
      integer i0,istep,itech,ncase
```

Step 2: **Discount rate (*disc*) is read.**

```
      read(i0,*)
      read(i0,*) disc
      disc=disc/100.0
```

Step 3: **Number of technology cases is read. Note that SRPM model considers only one technology (current technology); therefore, data entry for number of technology should be set to 1.**

```
      read(i0,*)
      read(i0,*) ncase
      call chkdim(ncase,qtech,'qtech')
```

Step 4: **Data entries for the cost data are read.**

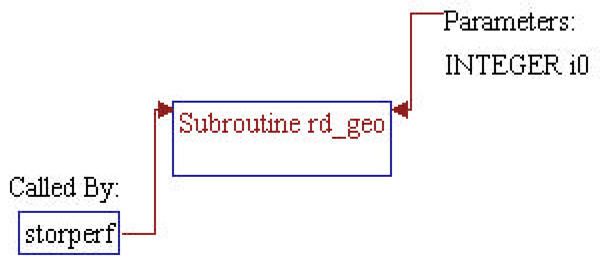
```
      do itech=1,ncase
        read(i0,*)
        read(i0,'(a)') casenm(itech)
        read(i0,*)
        read(i0,*) ewc_fac(itech)
        read(i0,*)
        read(i0,*) lbc_fac(itech)
        read(i0,*)
        read(i0,*) gg_fac(itech)
        read(i0,*)
        read(i0,*) pdry_dev(itech)
        pdry_dev(itech)=pdry_dev(itech)/100
        read(i0,*)
        read(i0,*) ewc_tan(itech)
        ewc_tan(itech)=ewc_tan(itech)/100
        read(i0,*)
        read(i0,*) dwc_tan(itech)
        dwc_tan(itech)=dwc_tan(itech)/100
        read(i0,*)
        read(i0,*) fac_tan(itech)
        fac_tan(itech)=fac_tan(itech)/100
        read(i0,*)
        read(i0,*) eccm(itech)
        read(i0,*)
        read(i0,*) ga_exp_m(itech)
        read(i0,*)
        read(i0,*) ga_cap_m(itech)
        read(i0,*)
        read(i0,*) ndwcreg(itech)
        call chkdim(ndwcreg(itech),qreg,'qreg')
        read(i0,*)
        do ireg=1,ndwcreg(itech)
          read(i0,*)
     &    dwc_reg(itech,ireg),dwck(itech,ireg),dwcx(itech,ireg),
     &    dwcxx(itech,ireg),dwcxxx(itech,ireg),dcstf(itech,ireg)
        enddo
        read(i0,*)
     &  dwc_reg(itech,qreg+1),dwck(itech,qreg+1),dwcx(itech,qreg+1),
     &  dwcxx(itech,qreg+1),dwcxxx(itech,qreg+1),dcstf(itech,qreg+1)
        read(i0,*)
        read(i0,*)newcreg(itech)
        read(i0,*)
        if(newcreg(itech).le.40)then
          call chkdim(newcreg(itech),qreg,'qreg')
          ienvr=qreg
        else
          call chkdim(newcreg(itech),qstate-1,'qstate')
          ienvr=newcreg(itech)
        endif
        do ireg=1,newcreg(itech)
          read(i0,*)ewc_reg(itech,ireg),
     &    env_et(itech,ireg),env_ei(itech,ireg),env_ee(itech,ireg),
     &    env_nt(itech,ireg),env_ni(itech,ireg),env_ne(itech,ireg),
     &    env_nf(itech,ireg),env_g(itech,ireg),env_w(itech,ireg)
        enddo
        read(i0,*)ewc_reg(itech,ienvr+1),
     &  env_et(itech,ienvr+1),env_ei(itech,ienvr+1),
     &  env_ee(itech,ienvr+1),env_nt(itech,ienvr+1),
     &  env_ni(itech,ienvr+1),
     &  env_ne(itech,ienvr+1),env_nf(itech,ienvr+1),
     &  env_g(itech,ienvr+1),env_w(itech,ienvr+1)
        read(i0,*)
        read(i0,*)
        read(i0,*) fac_n(itech)
        read(i0,*)
        read(i0,*)
        call chkdim(fac_n(itech),qstep,'qstep')
        do istep=1,fac_n(itech)
          read(i0,*) fac_max(istep,itech),fac_k(istep,itech),
     &    fac_s(istep,itech)
        enddo
        read(i0,*)
        read(i0,*) stimfac(itech)
        read(i0,*)
        read(i0,*) oam_h2o(itech)
        read(i0,*)
        read(i0,*) oam_gas(itech),oam_inc(itech)
        read(i0,*)
        read(i0,*) nc
        read(i0,*)
        do ir=1,nc
          read(i0,*) ireg,comp_vc(itech,ir)
        enddo
        read(i0,*)
        read(i0,*) ireg,comp_vc(itech,nc+1)
        read(i0,*)
        read(i0,*)
        read(i0,*) nreg_fx(itech)
        call chkdim(nreg_fx(itech),qreg,'qreg')
        do ireg=1,nreg_fx(itech)
          read(i0,*)
```

```
         read(i0,'(a,t5,i2)') fxoam_reg(itech,ireg),fxoam_n(itech,ireg)
         call chkdim(fxoam_n(itech,ireg),qstep,'qstep')
         read(i0,*)
         read(i0,*)
         do istep=1,fxoam_n(itech,ireg)
           read(i0,*) fxoam_max(istep,itech,ireg),
    &      fxoam_k(istep,itech,ireg),
    &      fxoam_s(istep,itech,ireg)
         enddo !istep
       enddo  !ireg
       read(i0,*)
       read(i0,'(a,t5,i2)')
    &  fxoam_reg(itech,qreg+1),fxoam_n(itech,qreg+1)
       call chkdim(fxoam_n(itech,qreg+1),qstep,'qstep')
       read(i0,*)
       read(i0,*)
       do istep=1,fxoam_n(itech,qreg+1)
         read(i0,*) fxoam_max(istep,itech,qreg+1),
    &    fxoam_k(istep,itech,qreg+1),
    &    fxoam_s(istep,itech,qreg+1)
       enddo !istep
     enddo !itech
100  format(t3,f10.0)
10   format(i3)
     return
     end
```

SUB-PROGRAM RD_GEO()

MAIN THEME:	Subroutine to read data of reservoir property distribution by pay grade.
READS:	GEOLOGY.DAT
CREATES:	None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameter of the subroutine:

- *io* Unit number of input file GEOLOGY.DAT

```
      subroutine rd_geo(i0)
```

Note: Include files and local variable.

```
      include 'dimen.h'
      include 'geology.h'
      integer i0
```

Step 2: **Reservoir property distributions are read.**

```
      read(i0,*)
      read(i0,*) nrestype
      if(nrestype.gt.qrestype) then
       write(6,*) 'Number of reservoir type exceeds maximum allowed'
       write(6,*) 'Program must be Recompiled'
       stop
      endif
      read(i0,*)
      read(i0,*)
      do irestype=1,nrestype
       do ipay=1,3
        read(i0,*) res_map(irestype),id,area_fac(ipay,irestype),
     &    por_fac(ipay,irestype),netpay_fac(ipay,irestype),
     &    h2osat_fac(ipay,irestype),perm_fac(ipay,irestype)
       enddo
      enddo
```

Note: Read default values into *qth+1* dimension

```
      do ipay=1,3
       read(i0,*) id,id,area_fac(ipay,qrestype+1),
     &   por_fac(ipay,qrestype+1),netpay_fac(ipay,qrestype+1),
     &   h2osat_fac(ipay,qrestype+1),perm_fac(ipay,qrestype+1)
      enddo
      return
      end
```

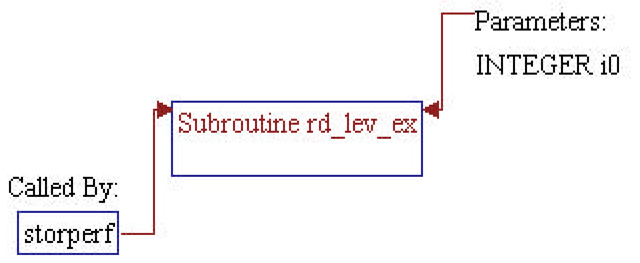
SUB-PROGRAM RD_LEV_EX()

MAIN THEME: Subroutine to read data of levelized costs and fixed and variable operating and maintenance (O&M) costs for existing storage reservoirs based on operating company.

READS: LEV.DAT

CREATES: None

ROUTINE INTERACTIONS:



Step 1: Subroutine declarations and definitions.

Note: Parameter of the subroutine:

- *io* Unit number of input file LEV.DAT

```
      subroutine  rd_lev_ex(i0)
```

Note: Include file.

```
      include 'storlp.h'
```

Step 2: Levelized costs and O&M costs are read.

```
      read(i0,*)
      read(i0,*)
      read(i0,*)
      read(i0,*)
      i = 1
  122 read(i0,*,end=124) icompany(i),lev_ex(i),fix_ex(i),var_ex(i)
      i = i+1
      goto 122
  124 n_tot_lev = i – 1
      return
      end
```

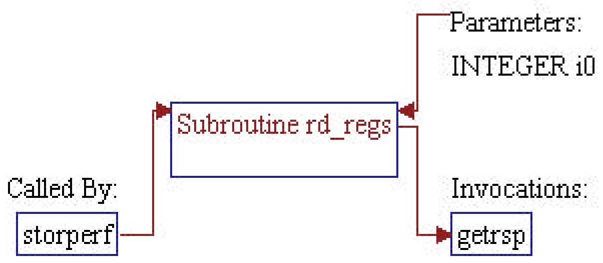
SUB-PROGRAM RD_REGS()

MAIN THEME: Subroutine to read list of *.STO files to be run through SRPM program and YES/NO flags for output files.

READS: REGIONS.DAT

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameter of the subroutine:

- *io* Unit number of input file REGIONS.DAT

```
      subroutine rd_regs(i0)
```

Note: Include files and local variables.

```
      include 'dimen.h'
      include 'global.h'
      character*3 ch3
      logical getrsp
```

Step 2: **YES/NO flags for output files and file names of *.STO files are read.**

```
      read(i0,*)
      read(i0,809) ch3
      l_tci=getrsp(ch3)
      read(i0,809) ch3
      l_pro=getrsp(ch3)
      read(i0,809) ch3
      l_tco=getrsp(ch3)
      read(i0,809) ch3
      l_prr=getrsp(ch3)
      read(i0,809) ch3
      l_npv=getrsp(ch3)
      read(i0,809) ch3
      l_prd=getrsp(ch3)
      ireg=1
      read(i0,*)
      read(i0,*)
 150  read(i0,'(a,t10,a,t34,a)',end=160) regnm(ireg),files(ireg),ch3
      runtype(ireg)=getrsp(ch3)
      ireg=ireg+1
      goto 150
 160  nreg=ireg-1
  809 format(t42,a3)
      return
      end
```

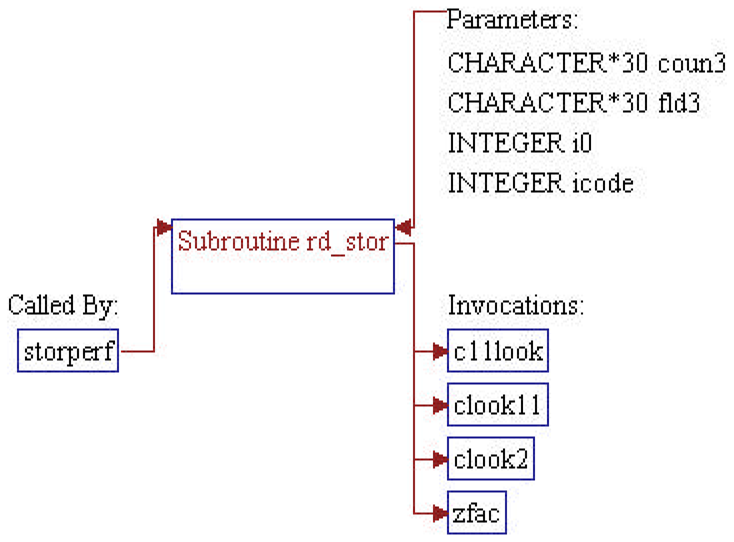
SUB-PROGRAM RD_STOR()

MAIN THEME: This routine reads one reservoir record from input file *.STO. Values of some reservoir properties are set to defaults and some of them are adjusted to match ultimate storage capacity reported by AGA. If requested, adjusted properties are used to overwrite database values.

READS: *.STO

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

Input:

• *io*	Unit number of input file *.STO

Output:

• *	Go to label condition for the calling routine: .TRUE. if end of file is encountered or .FALSE. otherwise.
• *icode*	Error code: 0 = no error, 1 = depth and pressure data not available, 2 = maximum deliverability and total working gas volume not exist in existing storage reservoir, 3 = not a gas storage reservoir (module # out of range), 4 = for existing storage and *iexruntyp=1* adjusted properties are not found or not available in input file ROCKPROP.ADJ, 5 = well spacing data based on regional average is required but not found in file DWLSPAC.DAT, 6 = unable to match AGA reported storage capacity
• *fld3*	30-character field name
• *coun3*	30-character reservoir name

```
      subroutine rd_stor(i0,*,icode,fld3,coun3)
```

Note: Include files and local variables.

```
      include 'rd_data.h'
      include 'gsamvar.h'
      include 'dimen.h'
      include 'tech.h'
      include 'global.h'
      include 'type5.h'
      character*30 fld3,coun3
```

Step 2: ***icode* is initialized to zero for no error.**

```
      icode = 0
```

Step 3: **Reading one reservoir record:**

- ***resid* is 11-character reservoir ID (GSAM ID)**
- ***icomp* is 3-digit company code (see LEV.DAT for the list of companies)**

- ***fld* is 30-character field name**
- ***resv* is 30-character reservoir name**
- ***stt* is 2-character state code**
- ***stid* is 4-digit state ID**
- ***yrdisc* is 4-digit discovery year (years)**
- ***yractv* is 4-digit year activated for storage (years)**
- ***pay* is pay thickness (feet)**
- ***maxdepth* is maximum depth of the reservoir (feet)**
- ***mindepth* is minimum depth of the reservour (feet)**
- ***pressure* is original pressure (psig)**
- ***acrelim* is approximate acreage reservoir limit (acre)**
- ***acretot* is approximate acreage total (acre)**
- ***iowells* is number of output and/or input wells**
- ***probwells* is number of pressure control and/or observation wells**
- ***compstat* is number of compressors**
- ***horspow* is horsepower of compressors (hp)**
- ***totbase* is base gas total volume (MMCF)**
- ***totwork* is working gas total volume (MMCF)**
- ***totfutr* is total future undeveloped or unused capacity (MMCF)**
- ***capacity* is ultimate storage capacity (MMCF)**
- ***maxdeliv* is designed maximum deliverability (MMCF/D)**
- ***pori* is porosity (%)**
- ***perm*i is permeability (md)**
- ***soil* is oil saturation (fraction)**
- ***sgas* is gas saturation (fraction)**
- ***swat* is water saturation (fraction)**
- ***apigrav* is gas api gravity (°API)**
- ***gasgrav* is gas specific gravity (dimensionless)**

```
                read(i0,10,end=100) resid,icomp,fld,resv,
             &    stt,stid,yrdisc,yractv,pay,maxdepth,
             &    mindepth,pressure,acrelim,acretot,iowells,
             &    probwells,compstat,horspow,totbase,totwork,
             &    totfutr,capacity,maxdeliv,pori,permi,soil,
             &    sgas,swat,apigrav,gasgrav
        10      format(1x,a11,1x,i3,1x,a30,1x,a33,1x,a2,3(1x,i4),1x,f6.1,
             &   2(1x,f7.1),1x,f6.1,2(1x,f7.1),2(1x,i5),1x,i4,1x,i5,
             &   4(1x,f8.1),1x,f7.2,1x,f4.1,1x,f6.1,3(1x,f4.2),
             &   1x,f4.1,1x,f5.3)
```

Step 4: **Store some parameters to working variables.**

Note: *gsamsr* is 2-digit ID for storage region
module is storage module number:
7 = depleted storage reservoir
8 = water aquiver storage

9 = salt cavern storage

statin is flag for existing/undeveloped storage reservoir:

0 = existing storage

1 = potential/undeveloped storage

rescod is 3-digit reservoir code

```
      gsamid = resid
      state = stid
      hp = horspow
      dbwells = iowells
      netpay = pay
      gasgrv = gasgrav
      watsat = swat
      gassat = sgas
      oilsat = soil
      fld3 = fld
      coun3 = resv
      read(gsamid(1:2),'(i2)') gsamsr
      read(gsamid(3:3),'(i1)') module
      modulesrpm = module
      read(gsamid(4:4),'(i1)') statin
      read(gsamid(9:11),'(i3)') rescod
      if (statin.le.0) statin = 0
      if (statin.gt.0) statin = 1
```

Step 5: **Return *icode=3* if *module* is out of range (not a storage reservoir).**

```
      if (module.lt.7.or.module.gt.9) then
        icode = 3
        return
      endif
```

Step 6: **If *maxdeliv* data is missing, calculate *maxdeliv* based on total working gas volume *totwork* and production period *prod_period*.**

```
      if (maxdeliv.le.0.0.and.totwork.ne.0.0)
   &    maxdeliv = totwork/(prod_period*365)
```

Step 7: **Return *icode=2* if existing reservoir does not have *maxdeliv* data. Note that *maxdeliv* will be used to adjust skin factor and permeability.**

```
      if (statin.eq.0.and.maxdeliv.le.0.0) then
        icode = 2
        return
      endif
```

Step 8: **Reservoir depth *depth* and initial pressure *presin* adjustments. Use hydrosatic gradient of *0.465 psi/ft* to obtain depth or pressure if one of these data is not available. If both depth and pressure data are not available, return *icode=1*.**

```
      depth = 0.0
      if (maxdepth.gt.0.0.and.mindepth.gt.0.0)
     &  depth = 0.5*(maxdepth+mindepth)
      if (maxdepth.le.0.0.and.mindepth.gt.0.0) depth = maxdepth
      if (maxdepth.gt.0.0.and.mindepth.le.0.0) depth = mindepth
      if (depth.le.0.0.and.pressure.gt.0.0)
     &  depth = (pressure-14.7)/0.465
      if (depth.gt.0.0.and.pressure.le.0.0)
     &  pressure = 1.1*(14.7+0.465*depth)
      if (pressure.le.0.0.and.depth.le.0.0) then
        icode = 1
        return
      endif
      presin = pressure
```

Step 9: **Minimum value of *yractv* is set to 1947. Number of economic years *nyrset* (used for potential/undeveloped reservoir) is set to one year.**

```
      if (yractv.le.1947) yractv = 1947
      nyrset = 1
```

Step 10: ***gasgrv* is set to 0.60 if data is not available. Concentration of impurities *h2s*, *co2,* and *n2* are initialized to zero. Bottomhole temperature *bhtemp* is calculated based on surface temperature of *60 °F* and temperature gradient of *0.014 °F/ft*.**

```
      if (gasgrv.le.0.0) gasgrv = 0.60
      h2s = 0.0
      co2 = 0.0
      n2 = 0.0
      bhtemp = 60+0.014*depth
```

Step 11: **Oil saturation is set to zero and gas and water saturations are normalized. If data for *gassat* or *watsat* is not available, default values are used (0.7 for gas and 0.3 for water).**

```
      totsat = gassat+watsat
      if (totsat.gt.0.0) then
        gassat = gassat/totsat
        watsat = watsat/totsat
      endif
```

```
        if (gassat.le.0.0) then
          gassat = 0.70
          watsat = 0.30
        endif
        if (watsat.le.0.0) then
          gassat = 0.70
          watsat = 0.30
        endif
```

Step 12: **Porosity *por* (fraction) and permeability *perm* (md) adjustments.**

Note: Use Timur correlation if porosity or permeability data is not available: k (md) = 1.360 * (por^4.4) / (swi^2), por and swi in % If both porosity and permeability data are not available, permeability value is set to 100 md and porosity is calculated based on Timur correlation.

```
        if (permi.le.0.0.and.pori.gt.0.0) then
          permi = 1.360*(pori**4.4)/(watsat*100.0)**2
        else if (permi.gt.0.0.and.pori.le.0.0) then
          pori = (permi*(watsat*100)**2/1.360)**0.2273
        else if (permi.le.0.0.and.pori.le.0.0) then
          permi = 100.0
          pori = (permi*(watsat*100)**2/1.360)**0.2273
        endif
```

Note: For salt cavern, porosity is at least 80% and permeability is at least 1000 md.

```
        if (module.eq.9) then
          pori = max(80.0,pori)
          permi= max(1000.0,permi)
        endif
```

Note: Store porosity and permeability to working variables *por* and *perm*.

```
        por = pori/100.0
        perm = permi
```

Step 13: **Drainage area *acprod* (acres) and well spacing *wlspac* (acres) calculations.**

Note: First *acprod* is set to *acrelim* if value for *acrelim* is greater than zero or it is set to *acretot*.

```
        acprod = acrelim
        if (acprod.le.0.1) acprod = acretot
```

Note: If *dbwells* is not available, *wlspac* is set to data of well spacings based on regional average assigned in file *DWLSPAC.DAT*. Return *icode=5* if well spacing data is not found in file *DWLSPAC.DAT*. If *acprod* is not available, *acprod* is set equal to four times the average well spacing. *dbwells* is then calculated based on *acprod* and *wlspac.*

```
if (dbwells.le.0.0) then
  icount = 0
  call clook2(gsamid(1:2),regname,n_tot_reg,icount)
  if (icount.eq.0) then
    icode = 5
    return
  endif
  wlspac = min_well(icount)
  if (acprod.le.0.0) acprod = wlspac*4.0
  dbwells = acprod/wlspac
```

Note: If value of *dbwells* is available, *wlspac* is calculated based on *acprod* and *dbwells*. For potential/undeveloped storage, values of *acprod* and *wlspac* are modified if permeability is greater than 200 md and number of wells is greater than 200. Calculations are based on data in file *DWLSPAC.DAT*.

```
else
  wlspac = acprod/dbwells
  if (statin.eq.1.and.permi.ge.200.and.dbwells.gt.200) then
    icount = 0
    call clook2(gsamid(1:2),regname,n_tot_reg,icount)
    if (icount.eq.0) then
      icode = 5
      return
    endif
    wlspac = min_well(icount)
    if (acprod.le.0.0) acprod = wlspac*4.0
    dbwells = acprod/wlspac
  endif
endif
```

Step 14: **Adjust well drainage area *acprod*, pay thickness *netpay*, porosity *por*, and well spacing *wlspac* to match total original gas in place *togip* with reported ultimate storage capacity *capacity.***

Note: *netpay* is set to 10 feet if data is not available.

```
if (netpay.le.0.0) netpay = 10.0
```

Note: Calculate *togip*.

```
bgi = 0.02829*zfac(gasgrv,presin,bhtemp)*(bhtemp+460)/presin
togip = 43560*acprod*netpay*por*gassat/bgi/1e06
```

Note: If *capacity* is not available, the calculated *togip* is used as *capacity*

```
if (capacity.le.0.0) capacity = togip
```

Note: Adjust the parameters based on the following adjustment factor:

0.4 for *acprod*
0.3 for *netpay*
0.15 for *por*
0.15 for *gassat*

Note: total of adjustment factors should be 1.0

```
      acprod = acprod*(capacity/togip)**(0.55)
      netpay = netpay*(capacity/togip)**(0.30)
      por = por*(capacity/togip)**(0.10)
      gassat = gassat*(capacity/togip)**(0.05)
      watsat = 1.0-gassat
      wlspac = acprod/dbwells
      togip = capacity
```

Note: For salt cavern, minimum pay thickness is set to 10 ft.

```
      if (module.eq.9) netpay = max(netpay,10.0)
```

Note: Final adjustment is done to the gas saturation based on the following constraints:

- *netpay* >= 10 ft.
- *wlspac* >= 20 acres.
- 0.05 <= *por* <= 0.25

```
      if (netpay.lt.10.0.or.wlspac.lt.20.0
   &    .or.por.lt.0.05.or.por.gt.0.25) then
        if (netpay.lt.10.0) netpay = 10.0
        if (wlspac.lt.20.0) then
          wlspac = 20.0
          acprod = dbwells*wlspac
        endif
        if (por.lt.0.05) por = 0.05
        if (por.gt.0.25) por = 0.25
        togip = 43560*acprod*netpay*por*gassat/bgi/1e06
        adjfac = capacity/togip
        gassat = gassat*adjfac
        if (gassat.lt.0.0.or.gassat.gt.1.0) then
          icode = 6
          return
        endif
        watsat = 1.0-gassat
        togip = 43560*acprod*netpay*por*gassat/bgi/1e06
      endif
```

Step 15: **For existing storage reservoir and *iexruntyp=1* (input file *SRPMSPEC.DAT* to be used to get adjusted properties: return *icode=4* if adjusted properties are not found or some values are not available in input file *ROCKPROP.ADJ*. *iadj* is pointer of current reservoir in adjusted properties arrays.**

```
      iadj = 0
      if (statin.eq.0.and.iexruntyp.eq.1) then
        call clook11(gsamid,gid,ncis,iadj)
        if (iadj.eq.0) then
```

```
            icode = 4
            return
          endif
          perm = permadj(iadj)
          por = poradj(iadj)
          gassat = sgadj(iadj)
          watsat = 1.0-gassat
          netpay = payadj(iadj)
          acprod = acpradj(iadj)
          wlspac = wlspadj(iadj)
          dbwells = nint(acprod/wlspac)
          acprod = dbwells*wlspac
          if (perm*por*netpay*acprod*wlspac.le.0.0) then
            icode = 4
            return
          endif
        endif
```

Step 16: ***perhor*** **is horizontal permeability (md)**
pervrt **is vertical permeability (md)**
permtx **is matrix permeability (md)**
portot **is total porosity (fraction)**
pormtx **is matrix porosity (fraction)**
welrad **is wellbore radius (ft)**

```
        perhor = perm
        pervrt = 0.3*perm
        permtx = 0.1*perm
        portot = por
        pormtx = max(0.04,portot-0.05)
        welrad = 0.354
        return
100     return 1
        end
```

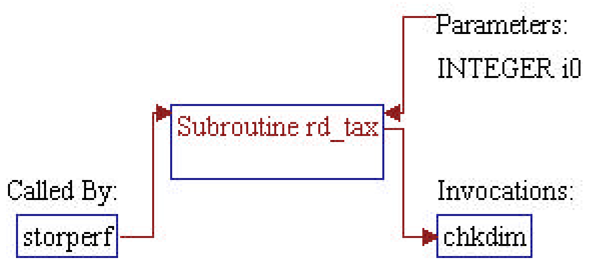
SUB-PROGRAM RD_TAX()

MAIN THEME: Subroutine to read state income taxes, oil and gas severance taxes, and ad-voleram taxes.

READS: TAXES.DAT

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameter of the subroutine:

- *io* Unit number of input file TAXES.DAT

```
      subroutine rd_tax(i0)
```

Note: Include files and local variables.

```
      include 'dimen.h'
      include 'tax_reg.h'
      integer i0
      real*4 tstate,toil,tgas,toil_p,tgas_p
```

Step 2: **Taxes data are read.**

```
      read(i0,*)
      read(i0,*)
      read(i0,*)
      read(i0,*) ntax_st
      call chkdim(ntax_st,qstate-1,'qstate')
      read(i0,*)
      read(i0,*)
      do istate=1,ntax_st
       read(i0,*) tax_st(istate),tstate,toil,toil_p,tgas,tgas_p
       strate(istate)=tstate/100.
       oil_sev(istate)   = toil/100.
       gas_sev(istate)   = tgas/100.
       gas_sev_p(istate) = tgas_p
       oil_sev_p(istate) = toil_p
      enddo
       read(i0,*) tax_st(qstate+1),tstate,toil,toil_p,tgas,tgas_p
      strate(qstate+1)=tstate/100.
        oil_sev(qstate+1)   = toil/100.
        gas_sev(qstate+1)   = tgas/100.
        gas_sev_p(qstate+1) = tgas_p
        oil_sev_p(qstate+1) = toil_p
      return
      end
```

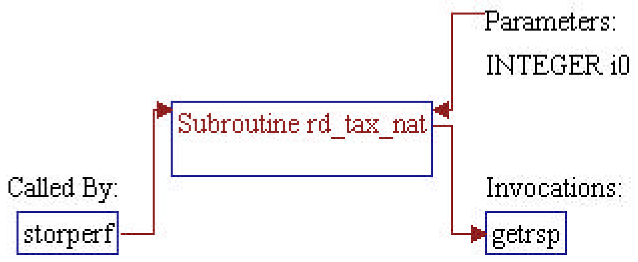
SUB-PROGRAM RD_TAX_NAT()

MAIN THEME: Subroutine to read data of generic tax structure (capitalize versus expense switches) assumptions.

READS: TAX_NAT.DAT

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameter of the subroutine:

- *io* Unit number of input file TAX_NAT.DAT

```
      subroutine rd_tax_nat(i0)
```

Note: Include files and local variables.

```
      include 'dimen.h'
      include 'tax_nat.h'
      include 'tax_reg.h'
      integer i0
      character*3 resp
      logical getrsp
```

Step 2: **Data of generic tax structure assumptions are read.**

```
      read(i0,*)
      read(i0,*) fedrate
      fedrate=fedrate/100
      read(i0,*)
      read(i0,*)ipdr
      ipdr=ipdr/100
      read(i0,*)
      read(i0,10)resp
      cidc=getrsp(resp)
      read(i0,*)
      read(i0,10) resp
      coi=getrsp(resp)
      read(i0,*)
      read(i0,10) resp
      envscn=getrsp(resp)
      read(i0,*)
      read(i0,10) resp
      ce=getrsp(resp)
      read(i0,*)
      read(i0,10)resp
      amt=getrsp(resp)
      read(i0,*)
      read(i0,10) resp
      credamt=getrsp(resp)
      read(i0,*)
      read(i0,*) smar
      smar=smar/100
      read(i0,*)
      read(i0,*)ipd
      ipd=ipd/100
      read(i0,*)
      read(i0,*)acer
      acer=acer/100
      read(i0,*)
      read(i0,*)ira
      ira=ira/100
      read(i0,*)
      read(i0,*)amtrate
      amtrate=amtrate/100
      read(i0,*)
```

```
read(i0,10) resp
eec=getrsp(resp)
read(i0,*)
read(i0,10) resp
nil=getrsp(resp)
read(i0,*)
read(i0,*) nill
nill=nill/100
read(i0,*)
read(i0,*)pdr
pdr=pdr/100
read(i0,*)
read(i0,*)piic
piic=piic/100
read(i0,*)
read(i0,*)eortcr
eortcr=eortcr/100
read(i0,*)
read(i0,10) resp
ggctc=getrsp(resp)
read(i0,*)
read(i0,*)ggctcr
ggctcr=ggctcr/100
read(i0,*)
read(i0,10) resp
ggetc=getrsp(resp)
read(i0,*)
read(i0,*)ggetcr
ggetcr=ggetcr/100
read(i0,*)
read(i0,10) resp
lactc=getrsp(resp)
read(i0,*)
read(i0,*)lactcr
lactcr=lactcr/100
read(i0,*)
read(i0,10) resp
laetc=getrsp(resp)
read(i0,*)
read(i0,*)laetcr
laetcr=laetcr/100
read(i0,*)
read(i0,10) resp
tdtc=getrsp(resp)
read(i0,*)
read(i0,*)tdtcr
tdtcr=tdtcr/100
read(i0,*)
read(i0,10) resp
idctc=getrsp(resp)
read(i0,*)
read(i0,*) idctcr
idctcr=idctcr/100
read(i0,*)
read(i0,10) resp
oitc=getrsp(resp)
read(i0,*)
read(i0,*)oitcr
oitcr=oitcr/100
read(i0,*)
read(i0,10) resp
ettc=getrsp(resp)
read(i0,*)
read(i0,*) ettcr
ettcr=ettcr/100
read(i0,*)
read(i0,10) resp
eitc=getrsp(resp)
read(i0,*)
read(i0,*)eitcr
eitcr=eitcr/100
```

```
      read(i0,*)
      read(i0,10) resp
      eoctc=getrsp(resp)
      read(i0,*)
      read(i0,*)eoctcr
      eoctcr=eoctcr/100
      read(i0,*)
      read(i0,10) resp
      tcoti=getrsp(resp)
      read(i0,*)
      read(i0,*)yr1
      read(i0,*)
      read(i0,10) resp
      tcoii=getrsp(resp)
      read(i0,*)
      read(i0,*)yr2
   10 format(a)
      read(i0,*)
      read(i0,*)
      read(i0,*)royrate
      royrate=royrate/100
      read(i0,*)
      read(i0,*)pggc
      pggc=pggc/100
      read(i0,*)
      read(i0,10) resp
      fsttax=getrsp(resp)
      read(i0,*)
      read(i0,*)yr3
      read(i0,*)
      read(i0,*)plac
      plac=plac/100
      return
      end
```

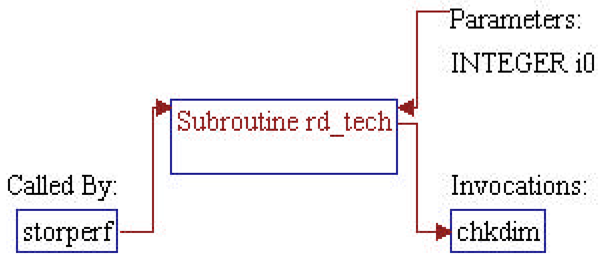
SUB-PROGRAM RD_TECH()

MAIN THEME: Subroutine to read data of technology specifications for various storage reservoir types.

READS: TECH.DAT

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameter of the subroutine:

- *io* Unit number of input file TECH.DAT

```
      subroutine rd_tech(i0)
```

Note: Include files and local variables.

```
      include 'dimen.h'
      include 'cost.h'
      include 'gsamvar.h'
      include 'tech.h'
      integer i0,npreg,npres,npay,nmod
```

Step 2: **Data of technology specifications are read.**

```
      read(i0,*)
      read(i0,*) ntech
      call chkdim(ntech,qtech,'qtech')
      do itech=1,ntech
       do ireg=1,qreg
        jtyp_tech(itech,ireg)=0
        enddo
       read(i0,*)
       read(i0,993) technm(itech)
  993  format(a20)
       read(i0,*)
       read(i0,*) prob_dry(itech)
       prob_dry(itech)=prob_dry(itech)/100
       read(i0,*)
       read(i0,*) wdtim_tech(itech)
       read(i0,*)
       read(i0,*) npreg
       call chkdim(npreg,qstate-1,'qstate')
       read(i0,*)
       do ireg=1,npreg
        read(i0,*)irxx,prorat_tech(itech,irxx)
       enddo
       read(i0,*)
        read(i0,*)ixxx,prorat_tech(itech,qstate)
      read(i0,*)
      read(i0,*) ntech_st
      call chkdim(ntech_st,qstate-1,'qstate')
      read(i0,*)
      do istate=1,ntech_st
       read(i0,*) tech_st(istate),proration(itech,istate)
      enddo
       read(i0,*)
       read(i0,*) npay
       call chkdim(npay,qreg,'qreg')
       read(i0,*)
       do ireg=1,npay
        read(i0,*)irxx,pay_tech(itech,irxx)
       enddo
       read(i0,*)
        read(i0,*)ixxx,pay_tech(itech,qreg)
       read(i0,*)
       read(i0,*) npres
```

```
           call chkdim(npres,qreg,'qreg')
           read(i0,*)
           do ireg=1,npres
            read(i0,*)irxx,psys_tech(itech,irxx)
           enddo
           read(i0,*)
            read(i0,*)ixxx,psys_tech(itech,qreg)
           read(i0,*)
           read(i0,*)nmod
          call chkdim(nmod,qrestype,'qrestype')
           read(i0,*)
           read(i0,*)(fracsk_tech(itech,imod),imod=1,nmod)
           read(i0,*)
           read(i0,*)(wrad_tech(itech,imod),imod=1,nmod)
           read(i0,*)
           read(i0,*)(fracxf_tech(itech,imod),imod=1,nmod)
           read(i0,*)
           read(i0,*)(fraccn_tech(itech,imod),imod=1,nmod)
          read(i0,*)
          read(i0,*)njreg
          call chkdim(njreg,qreg,'qreg')
          read(i0,*)
          do ireg=1,njreg
          read(i0,*)irxx,jtyp_tech(itech,irxx),jlen_tech(itech,irxx)
          enddo
          read(i0,*)
          read(i0,*)ndreg
          call chkdim(ndreg,qreg,'qreg')
          read(i0,*)
          do ireg=1,ndreg
          read(i0,*)irxx,diam_tech(itech,irxx)
          enddo
          read(i0,*)
          read(i0,*)irxx,diam_tech(itech,qreg)
          read(i0,*)
          enddo
          return
          end
```

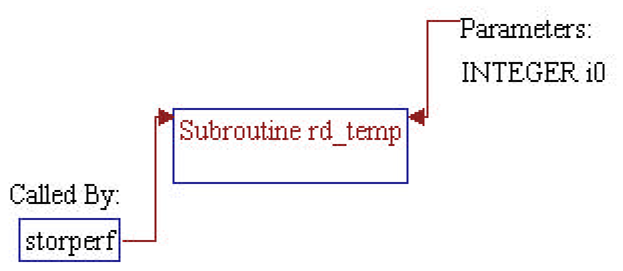
SUB-PROGRAM RD_TEMP()

MAIN THEME: Subroutine to read a template file used to generate type curve input parameters.

READS: TEMPLATE.DAT

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameter of the subroutine:

- *io* Unit number of input file TEMPLATE.DAT

```
          subroutine rd_temp(i0)
```

Note: Include files, common block, and local variables.

```
          include 'dimen.h'
          include 'gsamvar.h'
          integer iline,i0
          character*80 lines(qline)
          common/ddd/lines
```

Step 1: **String characters for the template are read.**

```
          iline=1
     10   read(i0,'(a80)',end=20) lines(iline)
           iline=iline+1
           if(iline.gt.qline) stop 4092
           goto 10
     20   continue
          return
          end
```

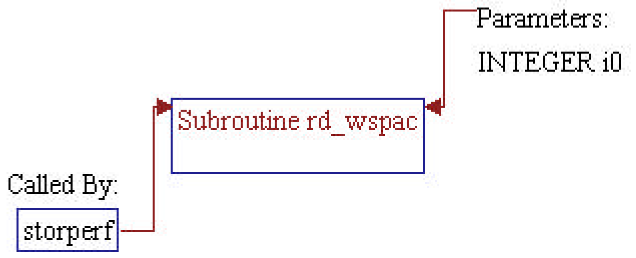
SUB-PROGRAM RD_WSPAC()

MAIN THEME: Subroutine to read data of minimum well spacing for existing/potential storage reservoirs as a function of storage/demand region.

READS: DWLSPAC.DAT

CREATES: None

ROUTINE INTERACTIONS:



Step 1: Subroutine declarations and definitions.

Note: Parameter of the subroutine:

- *io* Unit number of input file DWLSPAC.DAT.

```
      subroutine  rd_wspac(i0)
```

Note: Include files.

```
      include 'dimen.h'
      include 'rd_data.h'
      include 'gsamvar.h'
      include 'tech.h'
      include 'global.h'
```

Step 2: Well spacing data are read.

```
      read(i0,*)
      read(i0,*)
      read(i0,*)
      read(i0,*)
      i = 1
  122 read(i0,123,end=124) regname(i),min_well(i)
      i = i+1
      goto 122
  124 n_tot_reg = i – 1
  123 format(a2,t20,f6.0)
      return
      end
```

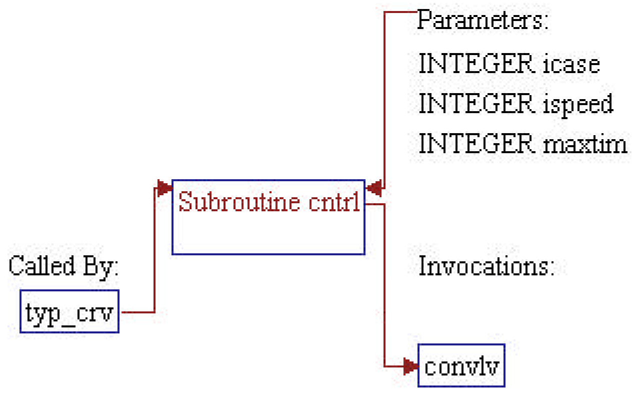
SUB-PROGRAM CNTRL()

MAIN THEME: This routine initializes pressures, rates, and well on/off variables.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *icase*	Case number (value should be 1 for no infill, no refrac)
• *ispeed*	Speedup option: 0=no speedup, 1=speedup (not currently used)
• *maxtim*	Number of time steps

```
      subroutine cntrl(icase,maxtim,ispeed)
```

Note: Include files.

```
      include 'type111.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type7.h'
      include 'type9.h'
      include 'type10.h'
```

Step 2: **Set up time steps**

```
      do i = 1,maxtim
        time(i) = deltat*i
      end do
```

Step 3: **Initialize pressures, rates, and well on/off variables. *j=1* (for primary wells only)**

```
      do i = 1,3
        do k = 1,maxtim
          j = 1
          pmin(i,j,k) = premin
          qg(i,j,k) = 0.
          cumgas(i,j,k) = 0.
          dq(i,j,k) = 0.
          caof(i,j,k) = 0.
          preavg(i,k) = 0.
          if (k.eq.1) preavg(i,k) = pinit(i)
        enddo
        kshut(i) = 0
      end do
      return
      end
```

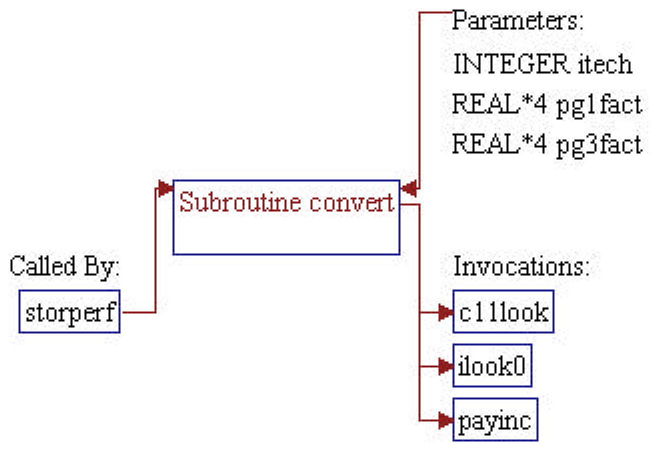
SUB-PROGRAM CONVERT()

MAIN THEME: This routine converts reservoir data read and set in subroutine RD_STOR() to type curve variables and distributes them on a pay grade level.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *itech*	Flag of technology: 1=current, 2=advanced (value should be 1 for SRPM)
• *pg1fact*	(not currently used)
• *pg3fact*	(not currently used)

```
      subroutine convert(itech,pg1fact,pg3fact)
```

Note: Include files and common blocks.

```
      include 'dimen.h'
      include 'global.h'
      include 'gsamvar.h'
      include 'field.h'
      include 'welldata.h'
      include 'geology.h'
      include 'type_out.h'
      include 'rd_data.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
      include 'tech.h'
      common /block1/ p1cum(3)
      common /skinvalue/ skinfac(3)
      common /scale/s_gip(3)
      common /num_well/ nwella(3)
```

Step 2: **Locate reservoir type pointer *irestype* in array *res_map* using subroutine *ILLOK0()*. Use default value *qrestype+1* if no match is found.**

```
      module = modulesrpm
      irestype = module
      call ilook0(module,res_map,nrestype,irestype)
      if (irestype.eq.0) irestype = qrestype+1
```

Step 3: **Store some properties to working variables:**

***gasgrv1* is gas specific gravity. Minimum value of 0.6 is used.**
***tem* is bottomhole temperature (°F)**
***cnch2s* is concentration of H2S (fraction)**

cncco2 **is concentration of CO2 (fraction)**
cncn2 **is concentration of N2 (fraction)**
wlspac1 **is well spacing (acre)**

```
gasgrv1 = max(gasgrv,0.60)
tem = bhtemp
cnch2s = h2s
cncco2 = co2
cncn2 = n2
wlspac1 = wlspac
```

Step 4: **Modify tubing diameter *diam* for depleted (*module=7*), water drive (*module=8*), and salt dome (*module=9*) reservoirs.**
Use 7.5" tubing diameter for depleted and water drive reservoirs.
Use 7.0" tubing diameter for salt dome reservoirs.

```
if(module.eq.7.or.module.eq.8) diam = 7.5
if(module.eq.9) diam = 7.0
```

Step 5: **Distribute some reservoir properties to pay grade level.**

```
do ipay = 1,3
  pinit(ipay) = presin
  perm(ipay) = perm_fac(ipay,irestype)*perhor
  permv(ipay) = perm_fac(ipay,irestype)*pervrt
  poros(ipay) = por_fac(ipay,irestype)*portot
  swi(ipay) = h2osat_fac(ipay,irestype)*watsat
  thick(ipay) = netpay_fac(ipay,irestype)*netpay
  permma(ipay) = perm_fac(ipay,irestype)*permtx
  porma(ipay) = por_fac(ipay,irestype)*pormtx
  area(ipay) = area_fac(ipay,irestype)*acprod
  depth1(ipay) = depth
  wspace(ipay) = wlspac1
enddo
```

Step 6: **Maximum value for initial water saturation in pay grade 2 is set to 99%.**

```
if (swi(2).eq.1.0) swi(2) = 0.99
```

Step 7: **Adjust the drainage area *area*, number of wells *nwella*, and thickness *thick* of each pay grade to balance gas in place with constraints number of wells in pay grades 1 and 3.**

Note: Calculate number of wells in pay grades 1 and 3 based on drainage area *area* and well spacing *wspace*. Values for number of wells are rounded to the nearest integer.

```
nwella(1) = nint(area(1)/wspace(1))
nwella(3) = nint(area(3)/wspace(3))
```

Note: Based on the calculated number of wells in pay grades 1 and 3, adjust the drainage area *area* in each pay grade. Residual area from pay grades 1 and 3 (due to rounding number of wells in these pay grades) is added to pay grade 2. Stop the program if the calculated drainage area in pay grade 2 is negative or zero.

```
      area(2) = area(2)+
     &   area(1)-wspace(1)*nwella(1)+
     &   area(3)-wspace(3)*nwella(3)
      area(1) = wspace(1)*nwella(1)
      area(3) = wspace(3)*nwella(3)
      if (area(2).le.0.0) then
        print*,'Calculation in subroutine CONVERT() found that ',
     &    'balanced drainage area of pay grade 2 is negative ',
     &    'or zero.'
        stop
      endif
```

Note: Based on the new drainage area, calculate number of wells in pay grade 2 (rounding the value to the nearest integer) and then recalculate well spacing to balance the *area*, *wspace*, and *nwella*.

```
      nwella(2) = nint(area(2)/wspace(2))
      wspace(2) = area(2)/nwella(2)
```

Note: Calculate gas in place for each pay grade based on drainage area *area*.

```
      do ipay = 1,3
        s_gip(ipay) = area(ipay)*thick(ipay)*
     &    poros(ipay)*(1.0-swi(ipay))
      enddo
```

Note: Residuals of gas in place in pay grades 1 and 3 (differences between gas in place based on *area* and gas in place based on *nwella*wspace*) are added to pay grade 1.

```
      s_gip(2) = s_gip(2)+
     &   s_gip(1)-(nwella(1)*wspace(1))*thick(1)*poros(1)*(1-swi(1))+
     &   s_gip(3)-(nwella(3)*wspace(3))*thick(3)*poros(3)*(1-swi(3))
```

Note: Pay thickness *thick* of pay grade 2 is adjusted based on gas in place balance.

```
      thick(2) = s_gip(2)/(area(2)*poros(2)*(1-swi(2)))
```

Step 8: **Determine well type *iwtype*:**
- *iwtype=0* for vertical wells (without hydraulic fracture).
- *iwtype=1* for horizontal wells.
- *iwtype=2* for hydraulically fractured vertical wells.

```
iwtype = 0
if (jtyp_tech(itech,gsamsr).ge.1) then
  iwtype = 1
else if (fracxf_tech(itech,module).gt.0.0) then
  iwtype = 2
endif
```

Step 9: **Assign well properties:**
- *welrad* is wellbore radius (feet)
- *rw* is wellbore radius (feet)
- *jtyp* is well type: 0=vertical, 1=horizontal
- *horlen* is horizontal well length (feet)
- *halfln* is fracture half length (feet)
- *cond* is fracture conductivity (md-ft)
- *skinfac* is skin factor

Note: Use wellbore radius from input file *TECH.DAT*

```
welrad = wrad_tech(itech,module)
```

Note: Assign the well properties separately for each pay grade

```
do i = 1,3
```

Note: *j=1* (for primary wells only).

```
  j = 1
```

Note: Store wellbore radius to working variable *rw*.

```
  rw(i,j) = welrad
```

Note: This is for vertical wells (without hydraulic fracture)

```
  if (iwtype.eq.0) then
    jtyp(i,j) = 0
    horlen(i,j) = 0.0
    halfln(i,j) = 0.0
    cond(i,j) = 0.0
    skinfac(i) = fracsk_tech(itech,module)
```

Note: This is for horizontal wells. Note that horizontal well is treated as infinite conductivity fracture. Therefore, conductivity is set to a big number (*1e6 md-ft*). Skin factor for horizontal wells is based on vertical well skin factor *fracsk_tech* and horizontal to vertical permeability ratio *perm/permv*.

```
  else if (iwtype.eq.1) then
```

```
                    jtyp(i,j) = 1
                    horlen(i,j) = jlen_tech(itech,gsamsr)
                    halfln(i,j) = horlen(i,j)
                    cond(i,j) = 1.0e6
                    fac_horz = (perm(i)/permv(i))**0.50
             &        *thick(i)/jlen_tech(itech,gsamsr)
                    skinfac(i) = fracsk_tech(itech,module)*fac_horz
```

Note: This is for hydraulically fractured vertical wells.

```
                  else
                    jtyp(i,j) = 0
                    horlen(i,j) = 0.0
                    halfln(i,j) = fracxf_tech(itech,module)
                    cond(i,j) = fraccn_tech(itech,module)
                    skinfac(i) = fracsk_tech(itech,module)
                  endif
                enddo
```

Step 10: **Use adjusted skin factor if the data is available in file *ROCKPROP.ADJ* and it is requested to be used in input file *SRPMSPEC.DAT* (only for existing storage reservoir).**

```
                if (statin.eq.0.and.iexruntyp.eq.1) then
                  skinfac(1) = skinadj(iadj)
                  skinfac(2) = skinadj(iadj)
                  skinfac(3) = skinadj(iadj)
                endif
```

Step 11: **Store skin factors to working variable *skin*. Note that SRPM model considers primary wells only.**

```
                do j = 1,3
                  skin(j,2,1) = skinfac(j)
                  skin(j,3,1) = skinfac(j)
                  skin(j,1,1) = skinfac(j)
                  skin(j,1,2) = skinfac(j)
                enddo
```

Step 12: **Determine maximum total allowable gas flow rate *ratmax* (MCF/D).**

***Note*:** For existing reservoir (*statin=0*) and non-technology run (*iexruntyp=0*): use *maxdeliv* (database maximum deliverability (MMCF/D)) as *ratmax*. *1000.0* is conversion from MMCF/D to MCF/D.

```
                if (statin.eq.0.and.iexruntyp.eq.0) then
                  ratmax = maxdeliv*1000.0
```

***Note*:** For others use proration specified in file *TECH.DAT* as *ratmax*, where:
- *0<prorat_tech<1* means *ratmax* should be set to *prorat_tech*aof*
- *prorat_tech>1* means *ratmax* should be set to *prorat_tech*
Note: *aof* is total gas flow rate based on absolute open flow.

```
            else
              ratmax = prorat_tech(itech,gsamsr)
```

***Note*:** Use default value if region proration is not available.

```
              if (ratmax.le.0.0) ratmax = prorat_tech(itech,qstate)
            endif
```

Step 13: **Assign minimum allowable wellhead pressure *premin*.**

```
            premin = psys_tech(itech,gsamsr)
```

***Note*:** Use default value if region minimum pressure is not available.

```
            if (premin.le.0.0) premin = psys_tech(itech,qreg)
```

Step 14: **Calculate average reservoir depth *avdep* and number of existing wells in each pay grade *io_wells* to be used in costing routines.**

```
            avdep = (depth+0.50*netpay)
            tot_area = area(1)+area(2)+area(3)
            do ipay = 1,3
              io_wells(ipay) = dbwells*area(ipay)/tot_area
            enddo
```

Step 15: **Convert SRPM module number (7,8, or 9) to:**
***-module=1* for vertical wells without hydraulic fracture**
***-module=2* for hydraulically fractured vertical wells or horizontal wells.**

```
            if (iwtype.eq.0) then
              module = 1
            else
              module = 2
            end if
```

Step 16: **Assign *module* to type curve variable *imod*.**
***j=1* (for primary wells only).**
Set minimum area to get type curve module running.

```
            do i = 1,3
              j = 1
              imod(i,j) = module
              if (area(i).le.0.0) area(i) = 0.0001
            enddo
            return
            end
```

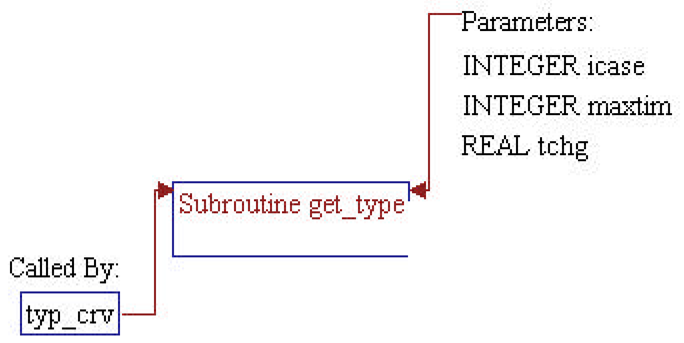
SUB-PROGRAM GET_TYPE()

MAIN THEME: Subroutine to get type curve output variables

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *maxtim*	Number of time steps
• *icase*	Case number (value should be 1 for no infill, no refrac)
• *tchg*	(not currently used)

```
      subroutine get_type(maxtim,icase,tchg)
```

Note: Include files, common block, and local variables.

```
      include 'dimen.h'
      include 'welldata.h'
      include 'type_out.h'
      include 'type111.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
      common /stchg/iwin_yr
      integer icase,nyr_ptr
      real*4 wells(3,3),frac1,frac2,prdinj,prdinj_n
      real*4 cyc_ptr,yr_ptr
```

Note: Calculate number of wells (primary wells only)

```
      do i=1,3
        wells(i,1) = area(i)/wspace(i)
        wells(i,2) = 0.
        wells(i,3) = 0.
      enddo
```

***Note*:** i: paygrade
j=1 (primary wells only)
k: maxtime
ogip1 is OGIP per well in a pay grade (MCF/Well)
type_ogip is total OGIP in a pay grade (BCF)

```
      do i = 1,3
        j = 1
        type_ogip(icase,i) = ogip1(i)*wells(i,1)/1e6
        type_well(icase,i) = type_well(icase,i)+wells(i,j)
```

***Note*:** calculate qg (mcf/day/well)

```
          do k=1,maxtim
            type_gas(icase,i,k) = type_gas(icase,i,k)+
     &        qg(i,j,k)*wells(i,j)*deltat*365.0/1.0e6
          enddo
        enddo
```

***Note*:** calculate pressures

```
        do k = 1,maxtim
          do i = 1,3
            type_pbhp(icase,i,k) = prbh(i,1,k)
            type_pwhp(icase,i,k) = prwh(i,1,k)
          enddo
        end do
        return
        end
```

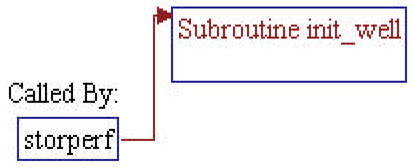
SUB-PROGRAM INIT_WELL()

MAIN THEME: Subroutine to initialize type curve variables.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

```
      subroutine init_well
```

Note: Include files and local variables.

```
      include 'dimen.h'
      include 'welldata.h'
      include 'type_out.h'
      include 'type3.h'
      include 'type1.h'
      include 'type2.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'cost.h'
      integer iyr,icase,ipay
```

Step 2: **Initialize type curve variables.**

```
      do icase=1,3
        do ipay=1,3
          do iyr=1,qyr
            type_gas(icase,ipay,iyr)=0.0
            type_pbhp(icase,ipay,iyr)=0.0
            type_pwhp(icase,ipay,iyr)=0.0
            type_ibhp(icase,ipay,iyr)=0.0
          enddo
          type_base(icase,ipay) =0.0
          type_work(icase,ipay) =0.0
          type_well(icase,ipay)=0.0
          type_ogip(icase,ipay)=0.0
        enddo
      enddo
      do ipay = 1,3
        pinit(ipay)= 0.0
        perm(ipay)=  0.0
        permv(ipay)= 0.0
        poros(ipay)= 0.0
        swi(ipay)=   0.0
        thick(ipay)= 0.0
        salin(ipay)= 0.0
        permma(ipay)= 0.0
        porma(ipay) = 0.0
        area(ipay)  = 0.0
        frcspc(ipay)= 0.0
        depth1(ipay)= 0.0
        wspace(ipay)= 0.0
        pl(ipay)    = 0.0
        tdes(ipay)  = 0.0
        gascon1(ipay)= 0.0
        rhoma(ipay)  = 0.0
        kuncon(ipay) = 0.0
        iloc(ipay)   = 0.0
      enddo
      return
      end
```

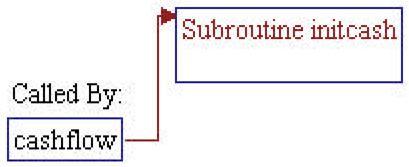
SUB-PROGRAM INITCASH ()

MAIN THEME: Subroutine to initialize cash flow variables

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

```
subroutine initcash
```

Note: Include files.

```
include 'dimen.h'
include 'cashflow.h'
integer iyr
```

Step 1: **Initialize cash flow variables.**

```
do iyr=1,qyr
  adjgross(iyr)=0.0
  netsales(iyr)=0.0
  toc(iyr)=0.0
  ga_exp(iyr)=0.0
  ga_cap(iyr)=0.0
  ii(iyr)=0.0
  intcap(iyr)=0.0
  ti(iyr)=0.0
  tci(iyr)=0.0
  tciadj(iyr)=0.0
  cap_base(iyr)=0.0
  depr(iyr)=0.0
  dggla(iyr)=0.0
  dep_crd(iyr)=0.0
  eggla(iyr)=0.0
  deplet(iyr)=0.0
  apd(iyr)=0.0
  nilb(iyr)=0.0
  eortca(iyr)=0.0
  idca(iyr)=0.0
  oia(iyr)=0.0
  iea(iyr)=0.0
  eoca(iyr)=0.0
  intadd(iyr)=0.0
  ggla(iyr)=0.0
  nibta(iyr)=0.0
  nibt(iyr)=0.0
  sttax(iyr)=0.0
  fti(iyr)=0.0
  fedtax(iyr)=0.0
  amti(iyr)=0.0
  acpamt(iyr)=0.0
  amint(iyr)=0.0
  ace(iyr)=0.0
  uamti(iyr)=0.0
  eidca(iyr)=0.0
  nifoag(iyr)=0.0
  dpidcs(iyr)=0.0
  idcpamt(iyr)=0.0
  aceadj(iyr)=0.0
  fedtaxc(iyr)=0.0
  niat(iyr)=0.0
  aatcf(iyr)=0.0
  datcf(iyr)=0.0
  catcf(iyr)=0.0
  sevtax(iyr)=0.0
  tfit(iyr)=0.0
  sfit(iyr)=0.0
  ucpamt(iyr)=0.0
```

```
            bamtp(iyr)=0.0
            lastyr=1
            intang_ewc(iyr)=0.0
            intang_dwc(iyr)=0.0
            tang_ewc(iyr)=0.0
            tang_dwc(iyr)=0.0
          enddo
          return
          end
```

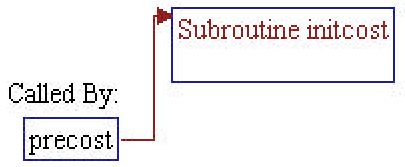
SUB-PROGRAM INITCOST ()

MAIN THEME: Subroutine to initialize costing variables.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

```
      subroutine initcost
```

Note: Include files and local variable.

```
      include 'dimen.h'
      include 'costing.h'
      integer iyr
```

Step 2: **Initialize costing variables.**

```
      do iyr=1,qyr
        icap(iyr)=0.0
        eicap(iyr)=0.0
        etcap(iyr)=0.0
        eoam(iyr)=0.0
        oam(iyr)=0.0
        inj(iyr)=0.0
        gravpen(iyr)=0.0
        transcst(iyr)=0.0
        gg(iyr)=0.0
        la(iyr)=0.0
        dwc(iyr)=0.0
        ewc(iyr)=0.0
        otc(iyr)=0.0
        stim(iyr)=0.0
        comp(iyr)=0.0
        recomp(iyr)=0.0
      enddo
      return
      end
```

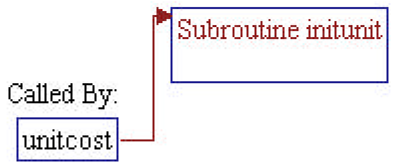
SUB-PROGRAM INITUNIT ()

MAIN THEME: Subroutine to initialize unit cost variables

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

```
subroutine initunit
```

Note: Include files and local variable.

```
include 'dimen.h'
include 'unitcost.h'
integer iyr
```

Step 2: **Initialize unit cost variables.**

```
ewc_w=0.0
dwc_w=0.0
stim_w=0.0
fac_w=0.0
env_cap_w=0.0
fxoam_w=0.0
voam_g=0.0
h2ooam_w=0.0
envni=0.0
envnt=0.0
envei=0.0
envet=0.0
env_oam_g=0.0
env_oam_w=0.0
env_oam_l=0.0
env_oam_n=0.0
lbc_frac=0.0
do iyr=1,qyr
  tang_m(iyr)=0.0
  intang_m(iyr)=0.0
  oam_m(iyr)=0.0
enddo
return
end
```

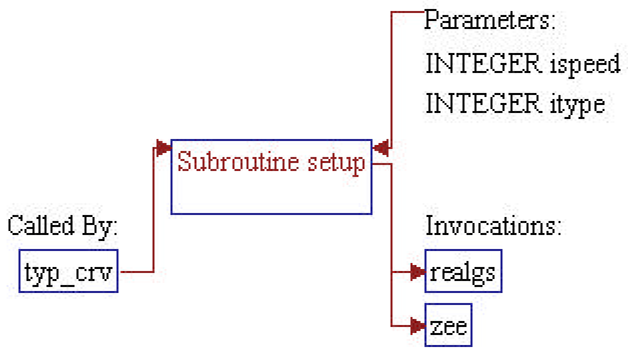
SUB-PROGRAM SETUP ()

MAIN THEME: Set up real gas potential, viscosity, and gas Z-factor arrays.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *ispeed*	Speedup flag: 0=no speedup, 1=speedup
• *itype*	Reservoir type (should be zero for conventional reservoir)

```
subroutine setup (ispeed,itype)
```

Note: Include files.

```
include 'type111.h'
include 'type1.h'
include 'type2.h'
include 'type3.h'
include 'type4.h'
include 'type5.h'
include 'type9.h'
include 'type10.h
```

Step 2: **Size of array is set.**

Note: Currently SRPM only utilize “no speedup” option (*ispeedup=0*).

```
if (ispeedup.eq.0) then
  narray = 99
else
  narray = 40
endif
```

Step 3: **Subroutine REALGS() is invoked to generate pressure functions.**

Note: Maximum pressure is set to 25% more than the highest initial pressures, to assure the ranges will be sufficient. The maximum pressure is forced to be at least 1000 psia.

```
pmax = max(pinit(1),pinit(2),pinit(3))*1.25
if (pmax.lt.1000.) pmax = 1000.
call realgs(pmax)
```

Step 3: **OGIP is calculated.**

```
do j = 1,3
  a = wspace(j)*43560.
```

```
                pi = pinit(j)
                zi = zee(pi,narray,preary,zary)
                ogip1(j) = a*thick(j)*poros(j)*520./(tem+460.)*
           &      (1.-swi(j))*pi/zi/14.7/1000.
              enddo
              return
              end
```

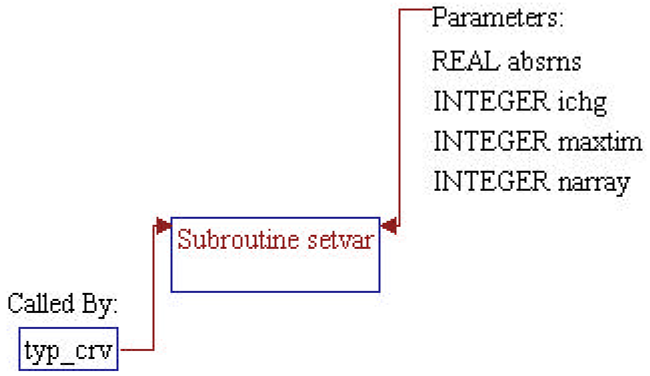
SUB-PROGRAM SETVAR ()

MAIN THEME: Initialize pressures, rates, absolute roughness of tubing, and number of data of pressure function arrays.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *absrns*	Absolute roughness of pipe, inches
• *maxtim*	Number of time steps
• *narray*	Size of array for pressure functions
• *ichg*	(not currently used)

```
      subroutine setvar(maxtim,narray,absrns,ichg)
```

Note: Include files.

```
      include 'dimen.h'
      include 'type111.h'
      include 'type5.h'
      include 'type9.h'
      include 'type10.h'
```

Step 2: **Size of array for pressure functions is defaulted to 99.**

```
      narray = 99
```

Step 3: **Absolute roughness is defaulted to 0.0006 inches.**

```
      absrns = 0.0006
```

Step 4: **Some pressure and rate arrays are initialized to zero.**

```
      do i = 1,3
        do j = 1,3
          do k = 1,maxtim
            pmin(i,j,k) = 0.0
            qmax(i,j,k) = 0.0
            prwh(i,j,k) = 0.0
            prbh(i,j,k) = 0.0
            qg(i,j,k) = 0.0
            dq(i,j,k) = 0.0
          enddo
        enddo
      enddo
      return
      end
```

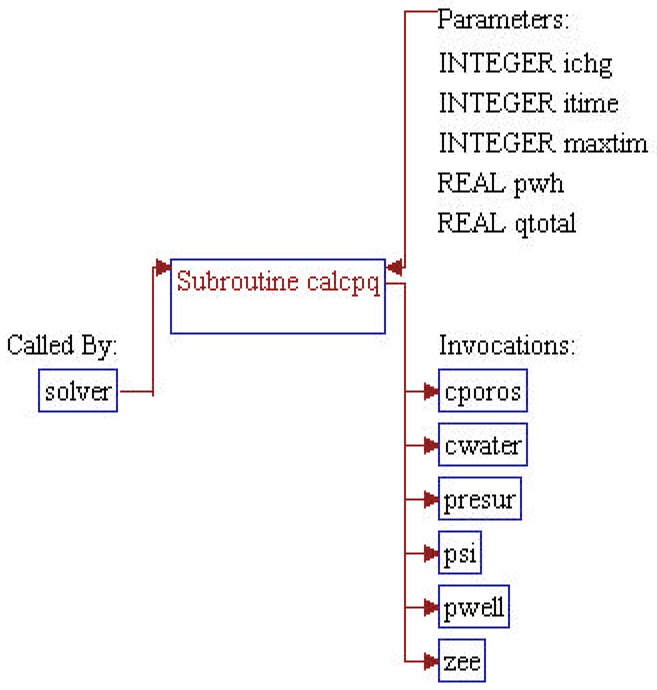
SUB-PROGRAM CALCPQ()

MAIN THEME: This routine computes wellhead and bottomhole pressures after rates have been determined.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

pwh	Wellhead pressure, psia
qtotal	Total production from field, mcfd (not currently used)
itime	Time step number
ichg	Flag to tell whether change has taken place (value should be 0 for primary wells only)
maxtim	Number of time steps(not currently used)

```
      subroutine calcpq (pwh,qtotal,itime,ichg,maxtim)
```

Note: Include files, common block, and local variables.

```
      include 'type111.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type9.h'
      include 'type10.h'
```

Step 2: **Stop production if total flow rate *q* or maximum possible flow rate *qmax3* is too low (less than 1 MCFD. Note that *qmax3* is determined based on maximum recovery efficiency *remax***

```
      if (ichg.ne.0) stop
```

Note: *dt* is time step from *time(itime-1)* to *time(itime)*. For first time step, use *time(1)*.

```
      dt = time(itime)
      if (itime.gt.1) dt = dt-time(itime-1)
      do j = 1,32
```

Note: Set initial pressure *pi*, minimum wellhead pressure *p1*, and average reservoir pressure *p2*. And calculate gas Z-factors at these pressures: *zi*, *z1*, and *z2*

```
        pi = pinit(j)
        zi = zee(pi,narray,preary,zary)
        p1 = premin
        z1 = zee(p1,narray,preary,zary)
        p2 = preavg(j,itime)
        z2 = zee(p1,narray,preary,zary)
```

Note: *remax* is calculated based on p/z at *p1*

```
          remax = 1.-(p1/z1)/(pi/zi)
```

Note: *q* is total flow rate. *qmax3* is maximum flow rate based on *remax*. Rates are in MCFD

```
          q = qg(j,1,itime)
          qmax3 = (ogip1(j)*remax-cumgas(j,1,itime))/(365.*dt)
```

Note: Shut in the wells if rates real low

```
          if ((q.lt.1.).or.(qmax3.lt.1.)) kshut(j) = 1
        end do
```

Step 3: Compute changes in gas rate *dq* and cumulative gas production *cumgas* for current time step.

```
        do i = 1,3
```

Note: *j=1* (for primary wells only).

```
          j = 1
          if (itime.eq.1) then
            dq(i,j,itime) = qg(i,j,itime)
            cumgas(i,j,itime)=qg(i,j,itime)*365.*time(1)
          else
            dq(i,j,itime)=qg(i,j,itime)-qg(i,j,itime-1)
            cumgas(i,j,itime) = cumgas(i,j,itime-1)+
     &        qg(i,j,itime)*365.*(time(itime)-time(itime-1))
          end if
        end do
```

Step 4: Calculate bottomhole pressure pbh and wellhead pressure *pwh*.

Note: *i* is loop for pay grade.
j=1 (for primary wells only).

```
        do i = 1,3
          j = 1
```

Note: Pseudo-pressure at current time *rgp* is calculated. First pseudo-pressure drop due to previous productions *dpsi* is subtracted from pseudo-pressure at initial pressure. Note that *dpsi* was calculated in subroutine *CONVLV()*.

```
          p = pinit(i)
          rgp = psi(p,narray,preary,psiary)-dpsi(i,j)
```

Note: Now subtract pseudo-pressure drops due to production and skin at current time step from *rgp*. The following variables were calculated in subroutine *CONVLV()*: *psicon(i)* is constant term of dimensionless flow rate for pay grade *i c(i,j,k)* is dimensionless pressure of well at location *j* due to production of well in location *k* for pay grade *i*: Note that in the SRPM model, only c(i,1,1) has value (no infill wells).

```
          q = qg(i,j,itime)
          s = skin(i,j,1)
          rgp = rgp-psicon(I)*(q*s+dq(i,1,itime)*c(i,j,1))
```

Note: Subroutine *PRESUR()* converts pseudo-pressure to pressure. Here the calculated pseudo-pressure at the wellbore *rgp* is converted to bottomhole pressure *pbh*.

```
          pbh = presur(rgp,narray,preary,psiary)
```

Note: In the following code, subroutine *PWELL()* is invoked to calculate wellhead pressure *pwh* given the bottomhole pressure *pbh* and flow rate *q*, where:

dep is depth of the reservoir.

ktyp is fluid production flag: (4) for water, anything else for gas.

```
          dep = depth1(i)
          ktyp = kuncon(i)
          call pwell(pwh,pbh,q,deriv,dep,2,ierr,ktyp,i)
```

Note: Values of bottomhole and wellhead pressures are stored.

```
          prbh(i,j,itime) = pbh
          prwh(i,j,itime) = pwh
        end do
```

Step 5: Calculate average reservoir pressure using bisection iteration

```
        do i = 1,3
```

Note: *fn* is objective function at initial pressure *pi*.

```
          cp = cporos(poros(i))
          pi = pinit(i)
          zi = zee(pi,narray,preary,zary)
          gp = cumgas(i,1,itime)+cumgas(i,2,itime)+
     &      cumgas(i,3,itime)*2.
          fn = pinit(i)/zi*(1.-gp/ogip1(i))
```

Note: *f* is objective function at first pressure entry in table *preary*.

```
       i1 = 1
       p = preary(i1)
       z = zary(i1)
       pavg = (pinit(i)+p)/2.
       cw = cwater(pavg,tem,salin(i))
       cwp = (cw*swi(i)+cp)/(1.-swi(i))
       f = p/z*(1.-cwp*(pi-p)-wed(i))
```

Note: Average reservoir pressure is less than first table entry (no iterative procedure is required): Using pressure in the first table entry, calculate *preavg* based on quadratic fit or set *preavg* to 14.7 if *fn* is zero or negative.

```
       if (f.gt.fn) then
        a = cwp
        b = 1.-cwp*pi-wed(i)+fn*(1.-z)/p
        disc = b*b+4*a*fn
        if (fn.gt.0.) then
         preavg(i,itime) = 2.*fn/(b+sqrt(disc))
        else
         preavg(i,itime) = 14.7
        end if
```

Note: Average reservoir pressure is greater than first table entry: Perform bisection iteration to get locations of two pressures in the table: *i0* and *i2*.

```
       else
        i0 = 1
        i2 = narray
        dn = float(narray)
        d = log(dn)/log(2.)+1.
        jmax = int(d)
        do j = 1,jmax
         if (i2-i0-1.gt.0) then
          i1 = (i2+i0)/2
          p = preary(i1)
          z = zary(i1)
          pavg = (pinit(i)+p)/2.
          cw = cwater(pavg,tem,salin(i))
          cwp = (cw*swi(i)+cp)/(1.-swi(i))
          f = p/z*(1.-cwp*(pi-p)-wed(i))
          if (f.ge.fn) then
           i2 = i1
          else
           i0 = i1
          end if
         end if
        end do
```

Note: Then calculate *preavg* using quadratic fit.

```
        cwp = (cw*swi(i)+cp)/(1.-swi(i))
        dzdp = (zary(i2)-zary(i0))/(preary(i2)-preary(i0))
        u = (zary(i0)-dzdp*preary(i0))*fn
        b = 1-cwp*pi-wed(i)-fn*dzdp
        disc= b*b+4*cwp*u
        preavg(i,itime)=2.*u/(b+sqrt(disc))
       end if
      end do
      return
      end
```

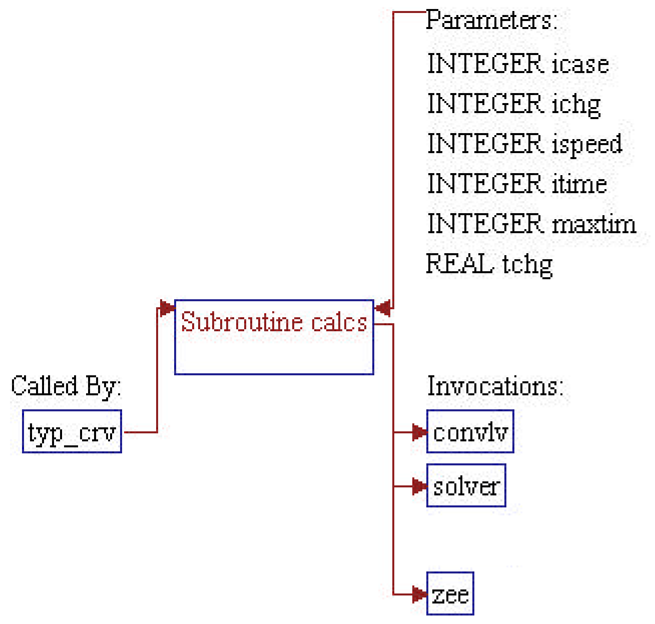
SUB-PROGRAM CALCS()

MAIN THEME: This routine calculates flow rates or pressures of the current time. Iterative procedure on average reservoir pressure is done where subroutine *CONVLV()* is used to calculate pressure drop due previous productions and subroutine *SOLVER()* is used to calculate flow rates or pressures.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

itime	Time step number
icase	Case number (value should be 1 for no infill, no refrac)
ichg	Flag to indicate whether development type change has taken place (value should be 0 for primary wells only)
tchg	Time at which automatic change in development type occurs (not used)
ispeed	Speedup option: 0=no speedup, 1=speedup
maxtim	Number of time steps

```
      subroutine calcs  (itime,icase,ichg,tchg,ispeed,maxtim)
```

Note: Include files, common block, and local variables.

```
      include 'type111.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type9.h'
      include 'type10.h'
      common /stchg/ iwin_yr
      dimension pguess(3),jshut(3)
```

Step 2: **Do not proceed if the time step number is higher than maximum number of time step.**

```
      if (icase.ne.1.and.ichg.ne.0) stop
      if (itime.gt.maxtim) return
```

Step 3: **If wells are shut in (*kshut=1*), use previous time step average reservoir pressure and cumulative gas production and return to the calling routine.**

```
      if ((kshut(1)+kshut(2)+kshut(3)).eq.3) then
        do i = 1,3
          preavg(i,itime) = preavg(i,itime-1)
          cumgas(i,1,itime) = cumgas(i,1,itime-1)
          cumgas(i,2,itime) = cumgas(i,2,itime-1)
          cumgas(i,3,itime) = cumgas(i,3,itime-1)
        end do
        return
      end if
```

Step 4: **Iteration process to solve for flow rates or pressures of current time is performed until the deviation between estimated and calculated average reservoir pressure *dp* within the specified tolerance *ptol*. Here the magnitude of *ptol* depends on the speedup option *ispeed*. Higher *ptol* is used for speedup mode.**

Note: *ptol* is pressure tolerance. *jmax* is maximum number of iteration. For speedup mode *ispeed=1*, use coarse pressure tolerance and less number of iteration.

```
      if (ispeed.eq.1) then
        ptol = 25.
        jmax = 3
      else
        ptol = 2.
        jmax = 13
      end if
```

Note: Before conducting the iteration, store original values of *kshut*, *tchg*, *ichg*, and *kshut*.

```
      tchg0 = -1.0
      ichg0 = 0
      do i = 1,3
        jshut(i) = kshut(i)
```

Note: Use initial reservoir pressure (for first time step) or use previous time step average reservoir pressure (for next time step) as the first estimated value of average reservoir pressure *pguess*.

```
        pguess(i) = pinit(i)
        if (itime.gt.1) pguess(i) = preavg(i,itime-1)
        preavg(i,itime) = pguess(i)
      end do
```

Note: *jtol* is iteration flag: 0=not done, 1=done.
First set to 0.

```
      jtol = 0
```

Note: Now iterate through subroutines *CONVLV()* and *SOLVER()* using successive substitution to solve for average reservoir pressure.

```
      do j = 1,jmax
        if (jtol.eq.0) then
```

Note: Set "kshut" to its original value

```
kshut(1) = jshut(1)
kshut(2) = jshut(2)
kshut(3) = jshut(3)
```

Note: Subroutine *CONVLV()* calculates pseudo-pressure drops at current time due to productions at previous time steps.

```
call convlv(itime,j,ispeed)
```

Note: Set *tchg* and *ichg* to their original values

```
tchg = tchg0
ichg = ichg0
```

Note: Subroutine *SOLVER()* solves for flow rates or pressures of current time.

```
call solver(itime,icase,ichg,tchg,ispeed,maxtim)
```

Note: *dp* is maximum deviation between estimated and calculated average reservoir pressures.

```
dp1 = pguess(1)-preavg(1,itime)
dp2 = pguess(2)-preavg(2,itime)
dp3 = pguess(3)-preavg(3,itime)
dp = max(abs(dp1),abs(dp2),abs(dp3))
```

Note: Use the calculated average reservoir pressure as estimated value for next iteration.

```
pguess(1) = preavg(1,itime)
pguess(2) = preavg(2,itime)
pguess(3) = preavg(3,itime)
```

Note: *ichk* tells whether calculated average reservoir is below the initial reservoir pressure or not. This is used to make sure that the iteration process is going to the right direction (decreasing in pressure due to production).

```
ichk = 0
do k = 1,3
  if (pguess(k).gt.pinit(k)) ichk = 1
end do
```

Note: Check for convergence:
Maximum pressure deviation is less or equal to pressure tolerance (*dp* <= *ptol*).
ichk=0 means reservoir pressure is depleting.

j>1 means to perform at least two iterations.
Set *jtol* to 1 if converged.

```
            if ((dp.le.ptol).and.(ichk.eq.0).and.(j.gt.1))
     &        jtol = 1
          end if
        end do
```

Note: Set *ichg* to its original value.

```
        ichg = ichg0
```

Note: Average reservoir pressure is set to the value at previous time step if flow rate is too low.

```
        do i = 1,3
          if (qg(i,1,itime).lt.1.)
     &      preavg(i,itime) = preavg(i,itime)-1.0
        end do
        return
        end
```

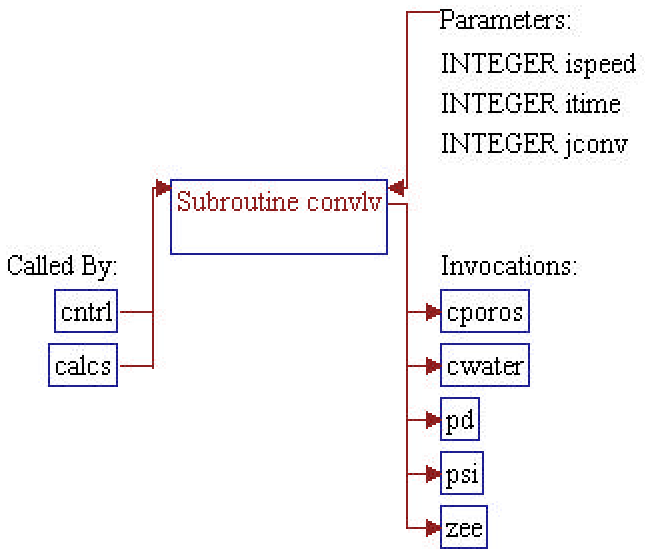
SUB-PROGRAM CONVLV()

MAIN THEME: This routine performs numerical convolution to determine pseudo-pressure drops caused by different flow rate in previous productions. The routine also calculates dimensionless pressures at the wellbore.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

itime	Time step number
iflag	Flag to tell whether this is first iteration or later
ispeed	Speedup option: 0=no speedup, 1=speedup

```
      subroutine convlv(itime,iflag,ispeed)
```

Note: Include files, common block, and local variables.

```
      include 'type111.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
```

Step 2: **Compute pressure drops in pay grades (*j=1,2,3*) of the reservoir. Note that current version of SRPM only considers one pay grade (pay grade #2).**

```
      do j = 1, 3
```

Step 3: **Calculate constant part of dimensionless time based on well drainage area *timcon*.**

Note: *pi* is initial reservoir pressure (psia)
psii is pseudo-pressure at *pi*.

```
        pi = pinit(j)
        zi = zee(pi,narray,preary,zary)
        psii = psi(pi,narray,preary,psiary)
```

Note: *pres* is average reservoir pressure. If average reservoir pressure is not yet available (i.e. for the first time step *itime=1* of the first iteration *iflag=1*), assume that the average reservoir pressure is 100 psi lower than the initial pressure.

```
        if (iflag.eq.1) then
          pres = pi-100.
          if (itime.gt.1) pres = preavg(j,itime-1)
          preavg(j,itime) = pres
        else
          pres = preavg(j,itime)
```

```
      end if
```

Note: *z* is gas Z-factor at *pres*.
psires is pseudo-pressure at *pres*.

```
      z = zee(pres,narray,preary,zary)
      psires = psi(pres,narray,preary,psiary)
```

Note: Since *pi* is always higher than *pres* (due to production), it is expected that *psii* is higher than *psires*. However, if the reverse is true, value of *psires* is forced to be lower than *psii* and *pres* and gas Z-factor are recalculated. This is done to avoid error in calculation of average compressibility-viscosity product *cmueff*.

```
      if (psii.le.psires) then
        psires = psii*.999999
        pres = presur(psires,narray,preary,psiary)
        z = zee(pres,narray,preary,zary)
      end if
```

Note: *cw* is water compressibility at average pressure *pavg*.
cp is pore volume compressibility.

```
      pavg = (pi+pres)/2.
      cw = cwater(pavg,tem,salin(j))
      cp = cporos(poros(j))
```

Note: Compute average compressibility-viscosity product *cmueff* based on material balance.

```
      ct = (cw*swi(j)+cp)/(1.-swi(j))
      cmueff = 2.*(1.-swi(j))/(psii-psires)*
 &      (pi/zi-pres/z*(1.-ct*(pi-pres)-wed(j)))
```

Note: Calculate constant part of dimensionless time based on well drainage area *timcon*.

- *0.006328* is time constant for days. The original time constant is 2.637E-4 for hours (see ERCB, 1975, pp.2-36) where:
  0.006328 = 2.637E-4 * 24
- *365* is conversion factor from years to days
- *wspace* is well spacing (acres) - *43560* is conversion factor from acres to ft2

```
      timcon(j) = 0.006328*perm(j)*365./
 &      (poros(j)*cmueff*wspace(j)*43560.)
```

Step 4: Calculate constant term of dimensionless flow rate

Note: - *1422* is rata constant for MCFD. The original rate constant is 1.422E6 for MMCFD (see ERCB, 1975, pp.2-36).

- *460* is conversion factor from deg. F to deg. R

```
      psicon(j) = 1422.*(tem+460.)/(perm(j)*thick(j))
```

Step 5: **Initialize pseudo-pressure drops.**

```
      dpsi(j,1) = 0.
      dpsi(j,2) = 0.
      dpsi(j,3) = 0.
```

Step 6: **Initialize loop for convolution.**

```
      do i = 1,itime
```

Step 7: **Determine time period *dt* in which the production in time step *i* interferes the production at current time step *itime*. Then calculate dimensionless time based on drainage area *tda*.**

```
        dt = time(itime)
        if (i.gt.1) dt = dt-time(i-1)
        tda = timcon(j)*dt
```

Step 8: **The following code is for horizontal well or hydraulically fractured vertical well. Calculate effective wellbore radius *rweff*, effective skin factor for horizontal well *shor*, etc.**

Note: *k=1* (for primary well only).

```
        k = 1
```

Note: Get reservoir module number.

```
        module = imod(j,k)
```

Note: First set effective wellbore radius as original wellbore radius.

```
        rweff = rw(j,k)
```

Note: Reservoir with hydraulic fracture or horizontal wells (*module=2 or 4*).

```
        if ((module.eq.2).or.(module.eq.4)) then
```

Step 9: **This is hydraulically fractured vertical well (*jtyp=0*)**

Note:
- *halfln* is the fracture half length
- *fcd* is dimensionless fracture conductivity

- *cond* is fracture conductivity (md-ft)
- Horizontal skin factor *shor* is set to zero
- *rweff* is set to the fracture half length

```
        if (jtyp(j,k).eq.0) then
          dlen = halfln(j,k)
          fcd = cond(j,k)/(perm(j)*dlen)
          if (fcd.le.0.) fcd = 100000.
          rweff = dlen
          shor = 0.
```

Note: This is a horizontal well (*jtyp=1*). NOTE: Horizontal well is treated as fractured vertical well by setting: - Infinite conductivity (*fcd* is set to 100000). - Effective wellbore radius (which is equal to fracture half length) is set to half of horizontal section of the horizontal well (*rweff* = *horlen/2*, where *horlen* is length of horizontal section of the horizontal well) - Effective skin factor *shor* is calculated accordingly

```
        else if (jtyp(j,k).eq.1) then
          dlen = horlen(j,k)
          fcd = 100000.
          rweff = dlen/2.
          ratio = sqrt(permv(j)/perm(j))
          rwd = rw(j,k)/thick(j)*ratio
          shor = -2.*thick(j)/dlen/ratio*
&            log(2.*asin(3.1415926*rwd))
        end if
      end if
```

Step 10: **Naturally fractured reservoirs (*module=3 or 4*): Calculate Warren parameters for subroutine *WARREN()*.**

```
      if ((module.eq.3).or.(module.eq.4)) then
        omega = porma(j)/poros(j)
        dlam = 12.*permma(j)/perm(j)*(rweff/frcspc(j))**2
      end if
```

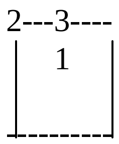
Step 11: **Subroutine *PD()* calculates dimensionless pressures at the well *pdw*. Dimensionless pressures for infill wells *pdcorn* and *pdedge*) are ignored.**

```
2---3----      (1) location of pdw
|   1   |      (2) location of pdcorn (first infills)
|       |      (3) location of pdedge (second infills)
|_______|
```

```
      arw = sqrt(wspace(j)*43560.)/rweff
      call pd(tda,module,arw,fcd,shor,omega,dlam,
&       ispeed,pdw,pdcorn,pdedge,ierr)
```

Note: Store dimensionless pressures to array *c* to be used to compute current flow rates and pressures, where:

c(j,k1,k2) is dimensionless pressure of well at location *k1* due to production of well in location *k2* for pay grade *j*:



Note that in the SRPM model, only c(j,1,1) has value (no infill wells).

```
                    c(j,k,k) = pdw
```

Note: *dpsi* is pressure drop due to production in previous time steps. The calculation for *dpsi* is done only for time step less than *itime.* Note that *dpsi* will only have value if there is a difference between flow rates in time step *i* and time step *itime.* Therefore, *dq* for the first time step is zero. The flow rate difference *dq* is calculated in subroutine *CALCPQ()*. The value of *dq* in the following equation is obtained from previous iteration level.

```
                    if (i.lt.itime) dpsi(j,k) = dpsi(j,k)+
             &        psicon(j)*dq(j,1,i)*c(j,k,1)
                  end do
                end do
                return
                end
```

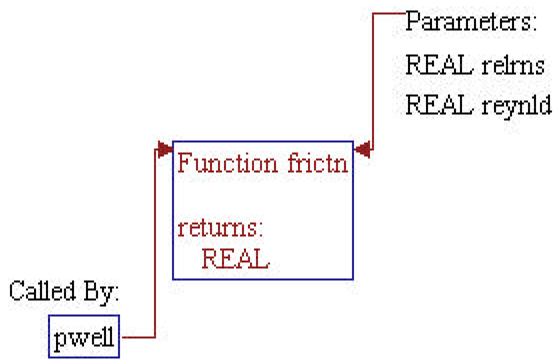
SUB-PROGRAM FRICTN()

MAIN THEME: This routine computes the Moody friction factor from Colebrook-White equation using Newton-Raphson iteration.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

reynld	Reynold number
relrns	Relative roughness, dimensionless

```
function frictn(reynld,relrns)
```

Step 2: **Return *frictn=1* if Reynold number *reynld<64*.**

```
if (reynld.lt.64.) then
  frictn = 1.
  return
end if
```

Step 3: **Use laminar formula for *reynld<2500*.**

```
if (reynld.lt.2500.) then
  frictn = 64./reynld
  return
end if
```

Step 4: **Use Colebrook-White formula for *reynld>2500*.**
Use 3 Newton-Raphson iterations on *x=1/f0.5*, starting at *f=1/36*. *fx* is the function to be solved, and *deriv* is the derivative of the function with respect to *x*.**

```
x = 6
do i = 1,3
  fx = x+0.868589*log(2.51/reynld*x+0.27*relrns)
  deriv = 1+0.868589/(x+0.27*relrns/2.51*reynld)
  x = x-fx/deriv
end do
frictn = 1./(x*x)
return
end
```

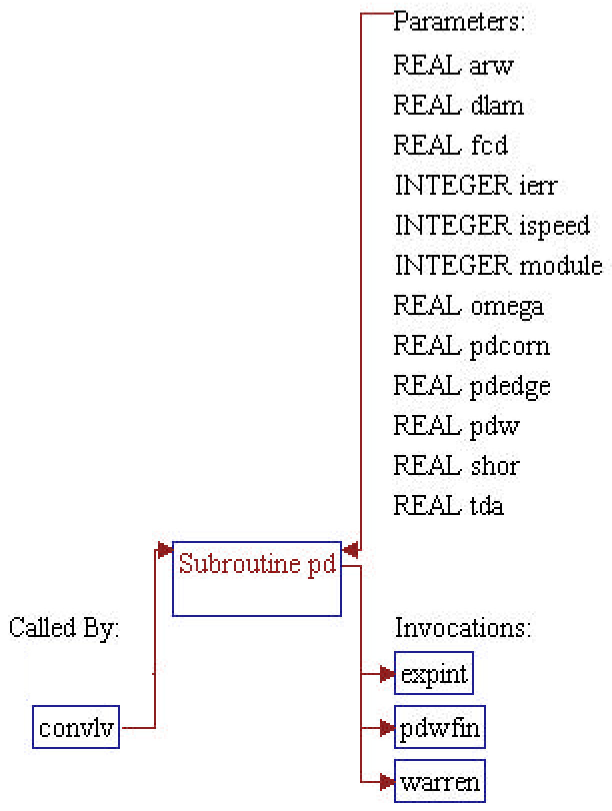
SUB-PROGRAM PD()

MAIN THEME: This routine computes dimensionless pressure based on radial flow to a well in the center of a closed, square reservoirs. For reservoir with hydraulic fracture or horizontal wells, subroutine *PDWFIN()* is invoked to calculate the effect of finite conductivity fracture to the dimensionless pressure. For naturally fractured reservoirs, subroutine *WARREN()* is invoked to calculate the effect of naturally fractured reservoir to the dimensionless pressure.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

tda	Dimensionless time based on well drainage radius
module	Reservoir module number
arw	Dimensionless area factor, a**0.5/rw, based on drainage area of primary well. xf/2 should be used instead of rw for fractured or horizontal wells, where xf is fracture half length
fcd	Dimensionless fracture conductivity, kfw/kxf for vertically fractured and horizontal wells (modules 2 and 4)
shor	Equivalent skin factor for a horizontal well
omega	Warren and Root porosity-compressibility ratio: omega = $(\text{phi-c})_{\text{fractures}} / (\text{phi-c})_{\text{total}}$
dlam	Warren and Root interporosity flow parameter: dlam = $12(\text{perm})_{\text{matrix}} / (\text{perm})_{\text{total}}$ * (rw/frac spacing)**2
ispeed	Speedup option: 0=no speedup, 1=speedup

```
      subroutine pd(tda,module,arw,fcd,shor,omega,dlam,ispeed,
     &               pdw,pdcorn,pdedge,ierr)
```

Step 2: **Initialize error flag *ierr* and dimensionless pressures.**

```
      ierr = 0
      pdw = 0.
```

Step 3: **First compute dimensionless pressure for reservoir module 1 (conventional reservoir).**

Note: *pdwinf* is a dimensionless pressure for a well in an infinite reservoir.

```
      a1 = 1./(arw*arw)/(4.*tda)
      pdwinf = 0.5*expint(a1)
```

Note: For *tda<0.05*, calculate dimensionless pressures using one set of image wells.

```
      if (tda .lt. 0.05) then
        a1 = 1./(4.*tda)/(arw*arw)
```

```
                 a2 = 1./(4.*tda)
                 pdw = 0.5*expint(a1)+2.0*expint(a2)
```

Note: For *tda>=0.05*, calculate dimensionless pressures using pseudo-steady state calculation.

```
               else
                 pi = 3.141592654
                 a = 4.0*pi*pi*tda
                 if (a.gt.20.) a = 20.0
                 u = exp(-a)
                 pdw = 2.*pi*tda-1.310533+log(arw)-2.0/pi*
             &      (u+u*u/2.+u**4/4.+u**5/5.)
               end if
```

Step 4: **Modify dimensionless pressure *pdw* for reservoir with hydraulic fracture or horizontal wells (*module=2 or 4*).**

```
               if ((module.eq.2).or.(module.eq.4)) then
                 tdxf = tda*arw*arw/4.
                 if (tdxf.le.0.) tdxf = 0.
```

Note: Subroutine *PDWFIN()* calculates the effect of finite conductivity fracture to the dimensionless pressure *p*.

```
                 p = pdwfin(tdxf,fcd,shor,ispeed)
```

Note: *pdw* is modified by adding *p* and subtracting *pdwinf* from the calculated *pdw* in module 1.

```
                 pdw = pdw+p-pdwinf
               end if
```

Step 5: **Modify dimensionless pressure *pdw* for naturally fractured reservoirs (*module=3 or 4*).**

```
               if ((module.eq.3).or.(module.eq.4)) then
                 tdw = tda*arw*arw/4.
                 if (tdw.le.0.) tdw = 0.
```

Note: Subroutine *WARREN()* calculates the effect of naturally fractured reservoir to the dimensionless pressure *pnatf*.

```
                 pnatf = warren(tdw,omega,dlam)
```

Note: *pdw* is modified by adding *pnatf* from the calculated *pdw* in module 1.

```
                 pdw = pdw+pnatf
               end if
```

Step 6: **Set error flag to one and dimensionless pressure to big number to indicate that dimensionless time based on well drainage radius *tda* is negative.**

```
      if (tda.le.0.) then
        ierr = 1
        pdw = 1000000.
      end if
      return
      end
```

SUB-PROGRAM PDWFIN()

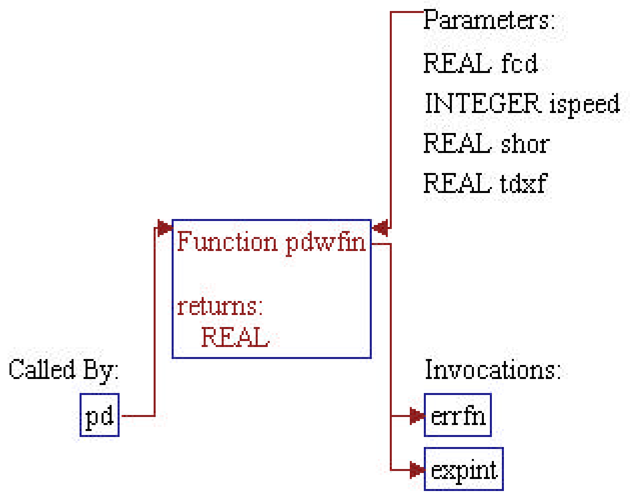
MAIN THEME: This routine computes dimensionless pressure for a well with a finite conductivity fracture producing at a constant rate in an infinite reservoir. The solution involves matching the pressure drop at a point along the fracture, using the uniform flux solution from Gringarten, et al (1974). The calculation point is based on a correlation from Blasingame and Poe.

Horizontal wells are computed based on an equivalent skin factor applied to the vertical fracture, using the analytical skin factor formula presented by Ozkan, Raghavan and Joshi, SPE Formation Evaluation, Dec., 1989, pp. 567-575.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

tdxf	Dimensionless time based on fracture half length xf, tdxf = kt/phi/mu/c/xf/xf
fcd	Dimensionless fracture conductivity, fcd = kfw/k/xf
shor	Equivalent skin factor for horizontal wells
ispeed	Speedup option: 0=no speedup, 1=speedup

```
      function pdwfin(tdxf,fcd,shor,ispeed)
```

Step 2: **Dimensionless pressure is calculated.**

```
      data a0,a1,a2,a3,a4,b0,b1,b2,b3,b4/
     &       0.759919, 0.465301, 0.562754, 0.363093, 0.029881,
     &       1.000000, 0.994770, 0.896679, 0.430707, 0.0467339/
      if (tdxf.le.0.0) tdxf = 0.0001
      f = fcd
      if (f.lt.0.5) f = 0.5
      if (f.gt.500.) f = 500.
      c = log(f)
      x = (a0+c*(a1+c*(a2+c*(a3+c*a4))))/
     &  (b0+c*(b1+c*(b2+c*(b3+c*b4))))
      arg1 = (1.-x)/2./sqrt(tdxf)
      arg2 = (1.+x)/2./sqrt(tdxf)
      arg3 = arg1*arg1
      arg4 = arg2*arg2
      c1 = errfn(arg1)
      c2 = errfn(arg2)
      c3 = expint(arg3)
      c4 = expint(arg4)
      pdwfin = (c1+c2)*sqrt(3.1415926*tdxf)/2.+
     &  (1.-x)*c3/4.+(1.+x)*c4/4.+shor
      return
      end
```

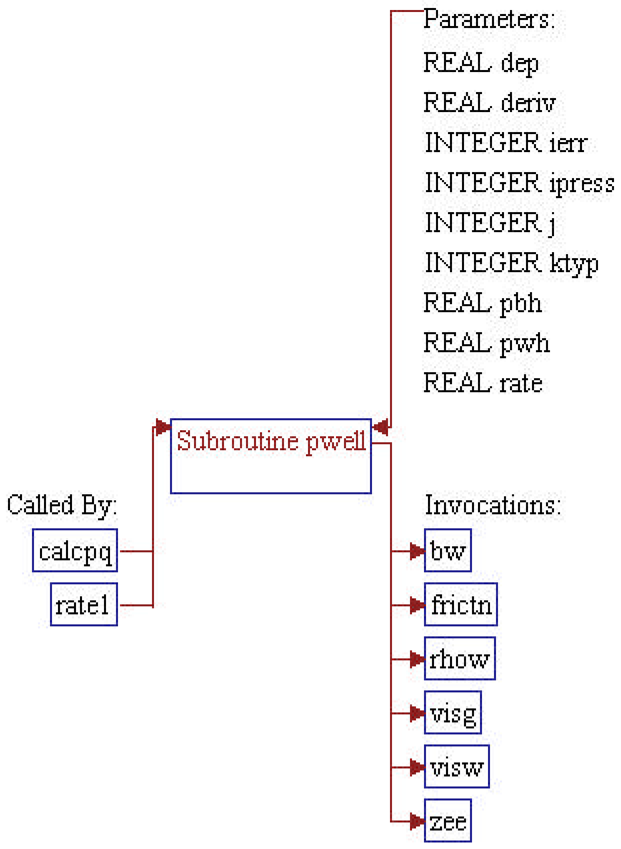
SUB-PROGRAM PWELL()

MAIN THEME: This routine calculates bottomhole pressure, or wellhead pressure, or gas flow rate depending the option *ipress*. The routine uses Smith's formula as presented by Katz, et al., pp. 309.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

itime	Time step number
pwh	Wellhead pressure, psia
pbh	Bottomhole pressure, psia
rate	Gas producing rate, MSCFD
deriv	Partial derivative of rate with respect to bottomhole pressure (used for rate solvers), mcfd/psia (return -1 if rate <= 0.)
dep	Reservoir depth, feet
ipress	Flag to define conditions: 1 = specified rate, pwh; find pbh 2 = specified rate, pbh; find pwh 3 = specified pwh, pbh; find rate
ktyp	Well type: 4 = water; anything else = gas
j	Pay grade number

```
      subroutine pwell(pwh,pbh,rate,deriv,dep,ipress,ierr,ktyp,j)
```

Note: Include files.

```
      include 'type111.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
```

Step 2: **Initialize *ierr* and *isolve*.**
***isolve* is convergence flag.**

```
      ierr = 0
      isolve = 0
```

Step 3: **To begin, assume fully turbulent flow and compute friction factor *fric* using Nikuradse's equation. The *fric* will be used as initial estimate of friction factor for flow of gas in production tubing.**

Note: *relrns* is relative roughness.

```
      relrns = absrns/diam
      fric = 1.0/(-4.60517*log(relrns)+1.14)**2
```

Step 4: **This is for water production cases (*ktyp=4*). Calculate *pwh,* or *pbh,* or *rate.***

```
      if (ktyp.eq.4) then
        if (ipress.eq.1) then
```

Note: Given *rate* and *pwh* calculate *pbh*. First, *pbh* is calculated based on hydrostatic head *dphead* assuming average pressure *pavg* equals to *pwh*.

```
          pavg = pwh
          dens = rhow(pavg,tem,salin(j))
          dphead = 0.433*dens*dep
          pbh = pwh+dphead
```

Note: Using the new *pbh*, calculate *dphead*, frictional pressure drop *dpf*, and then recalculate *pbh*.

```
          pavg = (pwh+pbh)/2.
          dens = rhow(pavg,tem,salin(j))
          wtrvis = visw(pavg,tem,salin(j))
          fvfw = bw(pavg,tem,salin(j))
          dens = rhow(pavg,tem,salin(j))
          reynld = 92.17*rate*fvfw*dens/(diam*wtrvis)
          fric = frictn(reynld,relrns)
          dpf = 1.338e-5*fric*dep/diam*dens*(rate*fvfw/diam**2)**2
          dphead = 0.433*dens*dep
          pbh = pwh+dphead+dpf
        else if (ipress.eq.2) then
```

Note: Given *rate* and *pbh* calculate *pwh*. First, *pwh* is calculated based on hydrostatic head *dphead* assuming average pressure *pavg* equals to half of *pbh*.

```
          pavg = pbh/2.
          dens = rhow(pavg,tem,salin(j))
          dphead = 0.433*dens*dep
          pwh = min(pbh-dphead,14.7)
```

Note: Using the new *pwh*, calculate *dphead*, frictional pressure drop *dpf*, and then recalculate *pwh*

```
          pavg = (pwh+pbh)/2.
          dens = rhow(pavg,tem,salin(j))
          wtrvis = visw(pavg,tem,salin(j))
          fvfw = bw(pavg,tem,salin(j))
          dens = rhow(pavg,tem,salin(j))
          reynld = 92.17*rate*fvfw*dens/(diam*wtrvis)
          fric = frictn(reynld,relrns)
          dpf = 1.338e-5*fric*dep/diam*dens*(rate*fvfw/diam**2)**2
          dphead = 0.433*dens*dep
          pwh = pbh-dphead-dpf
```

Note: Set *pwh=14.7* if the calculated *pwh* is less than 14.7 and assign a value of 1 to error flag *ierr*.

```
  if (pwh.le.14.7) then
   pwh = 14.7
   ierr = 1
  end if
```

Note: Given *pwh* and *pbh* calculate *rate.* Calculate hydrostatic head *dphead.*

```
 else
  pavg = (pwh+pbh)/2.
  dens = rhow(pavg,tem,salin(j))
  dphead = 0.433*dens*dep
```

Note: Use 1000 STB/D as initial estimate of *rate* if calculated bottomhole pressure is lower than the given *pbh*. Use rate of 1 STB/D otherwise.

```
  rate = 1000.
  if (pwh+dphead.ge.pbh) rate = 1.
```

Note: Set *rate=0* if *pbh-pwh* (which is total pressure drop) is less (or equal) to calculated *dphead*.

```
  f = pbh-pwh-dphead
  if (f.le.0.) then
   rate = 0.
   ierr = 1
```

Note: Iterate 3 times on *rate* to get correct friction factor.

```
  else
   reyc = 92.17*fvfw*dens/(diam*wtrvis)
   dpfc = 1.338e-5*dep/diam*dens*(fvfw/diam**2)**2
   do k = 1,3
    reynld = rate*reyc
    fric = frictn(reynld,relrns)
    dpf = rate**2*dpfc*fric
    rate = rate*sqrt(f/dpf)
   end do
  end if
 end if
```

Step 5: **This is for gas production cases (*ktyp*<>4). Calculate *pwh,* or *pbh,* or *rate.* Use initial guess for friction factor and recalculate as needed.**

```
else if (ipress.eq.1) then
```

Note: Given *rate* and *pwh* solve for *pbh*. Solve for *pbh* using a maximum of 50 bisection iterations with starting pressures of 0 and *pwh*. Return error code for no convergence and use last *pbh*.

```
          p1=0
          pbh = pwh
          do i = 1,50
            if (isolve.eq.0) then
              pavg = (pbh+pwh)/2.
              z = zee(pavg,narray,preary,zary)
              v = visg(pavg,narray,preary,visary,va)
              reynld = 20.06*rate*gasgrv1/(v*diam)
              fric = frictn(reynld,relrns)
              a = 0.03749*gasgrv1*dep/(tem+460.)/z
              ea = exp(a)
              pbh = sqrt(pwh**2*ea+(ea-1.)*
     &          (rate*(tem+460.)*z/38.377)**2*fric/diam**5)
              if (abs(pbh-p1).le.0.01) isolve = 1
              p1 = pbh
            end if
          end do
```

Note: Given *rate* and *pbh* solve for *pwh*. First check open flow of well *qof*. If *rate* > *qof* set *pwh=14.7*.

```
        else if (ipress.eq.2) then
          pwh = 14.7
          pavg = (pbh+pwh)/2.
          z = zee(pavg,narray,preary,zary)
          a = 0.03749*gasgrv1*dep/(tem+460.)/z
          ea = exp(a)
          u = pbh**2-ea*pwh**2
          qof = 0.
          if (u.gt.0.) qof = 38.377*diam**2.5/(tem+460.)/z*
     &      sqrt(u/(ea-1.))/sqrt(fric)
```

Note: Otherwise (*qof>rate*), solve for *pwh* using a maximum of 50 bisection iterations with starting pressures of *pbh* and *2*pbh* if *rate* is negative or starting pressures of *pbh* and *pbh/2* if *rate* is positive.

```
          if (qof.gt.rate) then
            p1 = 0.
            irate = 0
            if (rate.lt.0) then
              rate = -rate
              pwh = pbh*2.
              irate = 1
            else
              pwh = pbh/2.
            endif
            do i = 1,50
              if (isolve.eq.0) then
                pavg = (pbh+pwh)/2.
                z = zee(pavg,narray,preary,zary)
                v = visg(pavg,narray,preary,visary,va)
                reynld = 20.06*rate*gasgrv1/(v*diam)
                fric = frictn(reynld,relrns)
                a = 0.03749*gasgrv1*dep/(tem+460.)/z
                ea = exp(-a)
                if (irate.ne.1) then
                  pwh2 = (pbh**2*ea-(1.-ea)*
     &              (rate*(tem+460.)*z/38.377)**2*fric/diam**5)
                else
                  pwh2 = (pbh**2*ea-(ea-1.)*
     &              (rate*(tem+460.)*z/38.377)**2*fric/diam**5)
                endif
                if (pwh2.gt.1.) then
                  pwh = sqrt(pwh2)
```

```
           if (abs(pwh-p1).le.0.1) isolve=1
          else
           pwh = pwh/2.
          end if
          p1 = pwh
         end if
        end do
       end if
      else if (ipress .eq. 3) then
```

Note: Given *pwh* and *pbh* solve for *rate*. If pressure is less than static gradient then leave *rate=0* and *isolve=0*.

```
       rate = 0.
       pavg = (pbh+pwh)/2.
       z = zee(pavg,narray,preary,zary)
       v = visg(pavg,narray,preary,visary,va)
       a = 0.03749*gasgrv1*dep/(tem+460.)/z
       ea = exp(a)
       p1 = pwh*sqrt(ea)
```

Note: Perform maximum 50 iterations on friction factor to find *rate*

```
       if (abs(pbh-p1).gt.0.001) then
        u = abs(pbh**2-ea*pwh**2)/(ea-1)
        c = 38.377*diam**2.5/(tem+460.)/z*sqrt(u)
        q1 = c/sqrt(fric)
        do i = 1,50
         if (isolve.eq.0) then
          reynld = 20.06*q1*gasgrv1/(v*diam)
          fric = frictn(reynld,relrns)
          rate = c/sqrt(fric)
          if (abs(rate-q1).le.0.01) isolve=1
          q1 = rate
         end if
        end do
        if (pbh.le.p1) rate = -rate
       end if
      end if
```

Step 6: **Compute approximate derivative *deriv* which is the change of rate with respect to the change of bottomhole pressure for use with solution routines (for gas wells).**

```
      deriv = 1.e6
      if (isolve.eq.0) ierr = 1
      if (rate.gt.0.) then
       pavg = (pbh+pwh)/2.
       z  = zee(pavg,narray,preary,zary)
       p1 = pavg+1.
       z1 = zee(p1,narray,preary,zary)
       dz = (z1-z)/z
       a = 0.03749*gasgrv1*dep/(tem+460.)/z
       ea = exp(a)
       u = pbh**2-ea*pwh**2
       deriv = pbh*rate/u
      end if
      return
      end
```

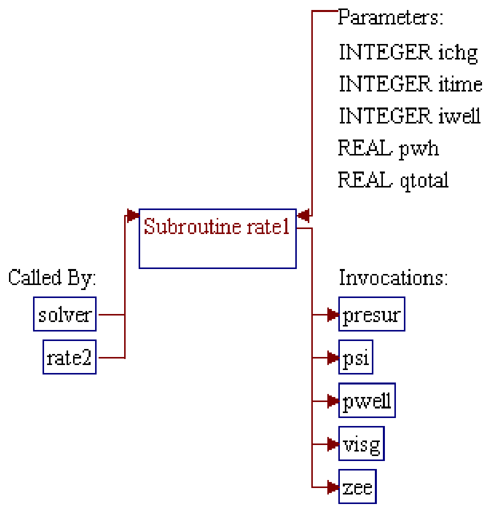
SUB-PROGRAM RATE1()

MAIN THEME: This routine calculates gas flow rate *qg* and total gas flow rate *qtotal* with a constraint of wellhead pressure *pwh* at time step *itime.*

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

pwh	Wellhead pressure, psia
itime	Time step number
ichg	Flag to tell whether change has taken place (value should be 0 for primary wells only)
iwell	Indicator for type of well (value should be 1 for primary wells)

```
      subroutine rate1(pwh,qtotal,itime,ichg,iwell)
```

Note: Include files.

```
      include 'type111.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type9.h'
      include 'type10.h'
```

Step 2: **Calculate flow rates for each pay grade separately. Initialize pay grade loop *j***

```
      if (ichg.ne.0) stop
      do j = 1, 3
```

Step 3: **Set gas flow rate *qg* to zero if the well is not on production (well is shut in *kshut>0*).**

```
        if (kshut(j).gt.0) then
          qg(j,iwell,itime) = 0.
```

Step 4: **Calculate gas flow rate if the well in on production. First assign skin factor and depth of the reservoir. *s* is skin factor of primary wells. *dep* is depth of the reservoir.**

```
        else
          s = skin(j,iwell,1)
          dep = depth1(j)
```

Step 5: **Subroutine *PWELL()* is invoked to calculate minimum bottomhole pressure *pbhmin* with constraints the specified wellhead pressure *pwh* and zero flow rate *qmin*. This is done to get hydrostatic pressure at the wellbore.**

```
qmin = 0.
ktyp = kuncon(j)
call pwell(pwh,pbhmin,qmin,deriv,dep,1,ierr,ktyp,j)
```

Step 6: **Subroutine *PSI()* is invoked to convert *pbhmin* to pseudo-pressure *psimin*.**

```
psimin = psi(pbhmin,narray,preary,psiary)
```

Step 7: **Determine maximum bottomhole pressure *pbhmax***

Note: First, convert initial pressure *pinit* to pseudo-pressure *psic*

```
psic = psi(pinit(j),narray,preary,psiary)
```

Note: Subtract pseudo-pressure drop due to previous productions *dpsi* (calculated in subroutine *CONVLV()*) from *psic* (only if the current time step is greater than one).

```
if (itime.gt.1) then
  psic = psic-dpsi(j,iwell)
```

Note: Then pseudo-pressure *psic* is modified with the effect of gas production at previous time step.
- *c* is dimensionless pressure at the wellbore which is *pdw* in subroutine *CONVLV()*.
- *psicon*qg* is dimensionless flow rate.

```
  psic=psic+psicon(j)*qg(j,1,itime-1)*c(j,iwell,1)
end if
```

Note: Now *psic* is a pseudo-pressure at the wellbore for previous time step and is used as a maximum bottomhole pressure by converting *psic* to *pbhmax* using subroutine *PRESUR()*.

```
pbhmax = presur(psic,narray,preary,psiary)
```

Step 8: **Set flow rate *q* equals to zero if *pbhmin*>*pbhmax*. This means there is no flow out from wellbore to the reservoir.**

```
      if (pbhmin.gt.pbhmax) then
        q = 0
```

Step 9: **Calculate maximum rate based on maximum bottomhole pressure *qmax1*.**

```
      else
        call pwell(pwh,pbhmax,qmax1,deriv,dep,3,ierr,ktyp,j)
```

Step 10: **Calculate maximum rate based on minimum bottomhole pseudo-pressure *qmax2*.**

```
        qmax2 = (psic-psimin)/(psicon(j)*(c(j,iwell,iwell)+s))
```

Step 11: **Calculate maximum rate based on maximum recovery *qmax3*.**

```
        dt = time(itime)
        if (itime.gt.1) dt = dt-time(itime-1)
        pi = pinit(j)
        zi = zee(pi,narray,preary,zary)
        p1 = premin
        z1 = zee(p1,narray,preary,zary)
        remax = 1.-(p1/z1)/(pi/zi)
        qmax3 = (ogip1(j)*remax-cumgas(j,1,itime))/(365.*dt)
```

Step 12: **Shut in the wells if *qmax3* is too low.**

```
        if (qmax3.le.1.) then
          kshut(j) = 1
          q = 0.
```

Step 13: **Maximum flow rate *qmx* is taken as the minimum between *qmax1*, *qmax2*, and *qmax3*. And set the value to be at least 1 MCFD.**

```
        else
          qmx = min(qmax1,qmax2,qmax3)
          qmx = max(qmx,1.)
```

Step 14: **Using the calculated *qmx* and the specified *pwd*, recalculate maximum bottomhole pressure *pbhmax* using subroutine *PWELL()*. Then convert *pbhmax* to pseudo-pressure *psimax*.**

Force *psimin* value to be 0.99 of *psimax* if *psimin* is higher than *psimax*.

```
      call pwell(pwh,pbhmax,qmx,deriv,dep,1,ierr,ktyp,j)
      psimax = psi(pbhmax,narray,preary,psiary)
      if (psimin.ge.psimax) psimin = psimax*0.99
```

Step 15: **Using quadratic fit of bottomhole pressure versus flow rate, determine flow rate *q* for initial guess in the iterative procedure to solve for flow rate *q* at the specified *pwh*.**

```
      a = psimin
      b =(psimax-a)/qmx**2
      aq = b
      bq = psicon(j)*(c(j,iwell,iwell)+s)
      cq = a-psic
      disc = bq**2-4.*aq*cq
      disc = max(disc,0.)
      sd = sqrt(disc)
      if ((200.*aq).lt.(sd-bq)) then
        q = ( sd - bq ) / (2. * aq)
      else
        q = 100.
      end if
      if (q.ge.0.0.and.qmx.ge.0.0) then
        q = max(q,qmx)
      else
        q = min(q,qmx)
      endif
```

Step 16: **Solve for flow rate *q* using Newton-Raphson iterative procedure. Convergence criterion of 0.01 MCFD is used.**

```
      isolve = 0
      do iter = 1, 100
        if (isolve.eq.0) then
          psi1 = psic-psicon(j)*(c(j,iwell,iwell)+s)*q
          pbh1 = presur(psi1,narray,preary,psiary)
          call pwell(pwh,pbh2,q,deriv,dep,1,ierr,ktyp,j)
          psi2 = psi(pbh2,narray,preary,psiary)
          f = psi2-psi1
          dp = pbh1-pbh2
          z = zee(pbh2,narray,preary,zary)
          v = visg(pbh2,narray,preary,visary,va)
          fp = 2.*pbh2/(v*z)/deriv+psicon(j)*
     &        (c(j,iwell,iwell)+s)
          delrat = f/fp
          if (abs(delrat).lt.0.01) then
            isolve = iter
          else
            qt = q-delrat
            if (qt.lt.q/2.) qt = q/2.
            if (qt.gt.q*2.)  qt = q*2.
            if (qt.gt.qmax3) qt = qmax3
            q = qt
          end if
        end if
      end do
```

```
            end if
          end if
```

Step 17: **Store the calculated *q* to variable gas production rate *qg*. Also calculate total gas flow rate *qtotal*.**

```
          qg(j,iwell,itime) = q
        end if
      end do
      qtotal = 0.
      do i = 1,3
        qtotal = qtotal+area(i)/wspace(i)*qg(i,1,itime)
      end do
      return
      end
```

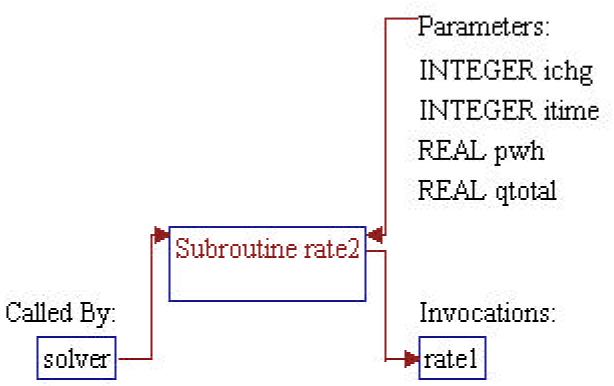
SUB-PROGRAM RATE2()

MAIN THEME: This routine calculates wellhead pressure *pwh* with a constraint of maximum total gas rate *ratmax* calculated in subroutine *SOLVER()*. The returned value of "*total* from this routine is expected to be very close to the value of *ratmax*.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

itime	Time step number
ichg	Flag to tell whether change has taken place (value should be 0 for primary wells only)

```
1               subroutine rate2(pwh,qtotal,itime,ichg)
```

Note: Include files and local variable.

```
2               include 'type111.h'
3               include 'type1.h'
4               include 'type2.h'
5               include 'type3.h'
6               include 'type4.h'
7               include 'type5.h'
8               include 'type6.h'
9               include 'type7.h'
10              include 'type9.h'
11              include 'type10.h'
12              dimension jshut(3)
```

Step 2: **Get pressures *p1* and *p2* for bisection iteration.**

```
14              pmax = max(pinit(1),pinit(2),pinit(3))
15              if (itime.gt.1)
16           &    pmax = max(preavg(1,itime-1),preavg(2,itime-1),
17           &      preavg(3,itime-1))
18              pstart = premin
19              p1 = pmax
20              p2 = pstart
```

Step 3: **Store original values of *kshut* before iteration.**

```
13              if (ichg.ne.0) stop
21              jshut(1) = kshut(1)
22              jshut(2) = kshut(2)
23              jshut(3) = kshut(3)
```

Step 4: **Perform a maximum of 3 bisection iterations to solve for wellhead pressure *p* to be used as initial guess pressure of Newton-Raphson iteration.**

```
24              do iter = 1,3
25                kshut(1) = jshut(1)
26                kshut(2) = jshut(2)
27                kshut(3) = jshut(3)
28                p = (pstart+pmax)/2.
```

```
        iwell = 1
        call rate1(p,qtotal,itime,ichg,iwell)
        f = qtotal - ratmax
        if (f.gt.0.) then
          pstart = p
        else
          pmax = p
        end if
      end do
```

Step 5: **Perform Newton-Raphson iteration to solve for wellhead pressure *p* with initial guess obtained from bisection iteration above.**

```
      p = (pstart+pmax)/2.
      delp = 2.
      isolve = 0
      do iter = 1,10
        if (isolve.eq.0) then
          kshut(1) = jshut(1)
          kshut(2) = jshut(2)
          kshut(3) = jshut(3)
          iwell = 1
          call rate1(p,qtotal,itime,ichg,iwell)
          f = qtotal-ratmax
          if (abs(f).lt.0.1) then
            isolve=iter
          else
            p1 = p+delp
            iwell = 1
            call rate1(p1,qtot1,itime,ichg,iwell)
            fp = qtot1-ratmax-f
            if (abs(fp).gt.0.0001) then
              pg = p-f/fp*delp
              if (pg.lt.p/2.) pg = p/2.
              if (pg.gt.pmax) pg = (p+pmax)/2.
              if (abs(fp).lt.0.02) delp =min(2.*delp,16.)
              p = pg
            end if
          end if
        end if
      end do
```

Step 6: **Store *p* to wellhead pressure variable *pwh*.**

```
      pwh = p
      return
      end
```

SUB-PROGRAM SOLVER()

MAIN THEME: This routine calculates flow rates or pressure of current time based on one of the following constraints:

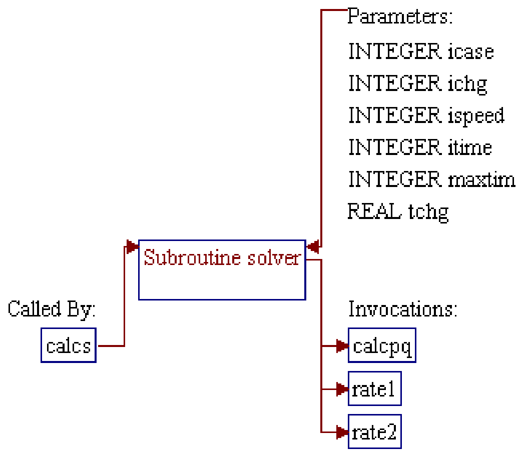
- Minimum allowable wellhead pressure constraint *premin* (if total gas rate is less than maximum allowable rate *ratmax*)
- Maximum allowable rate *ratmax* (if total gas rate based on pressure constraint is higher than maximum allowable rate *ratmax*).

Prior to the flow rates or pressure calculation, absolute open flow at current time is calculated based on a wellhead pressure of 14.7 psia. The value of *ratmax* is also assigned in this routine by utilizing the total gas flow based on the absolute open flow at the first time step.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

itime	Time step number
icase	Case number (value should be 1 for no infill, no refrac)
ichg	Flag to tell whether change has taken place (value should be 0 for primary wells only)
tchg	Time at which automatic change occurs (not used)
ispeed	Speedup option: 0=no speedup, 1=speedup
maxtim	Number of time steps (not used)

```
      subroutine solver(itime,icase,ichg,tchg,ispeed,maxtim)
```

Note: Include files and common block.

```
      include 'type111.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
      common /stchg/iwin_yr
```

Step 2: **Calculate absolute open flow *caof* and total gas flow rate *qtotal*:**

- **Set wellhead pressure *pwh* to 14.7 psia.**
- **Invoke subroutine *RATE1()* to calculate gas rates *qg*. *iwell=1* is a flag for primary well.**
- **Use *qg* as absolute open flow *caof*.**

```
      if (icase.ne.1.and.ichg.ne.0) stop
      pwh = 14.7
      iwell = 1
      call rate1(pwh,qtotal,itime,ichg,iwell)
      do i = 1,3
        caof(i,1,itime) = qg(i,1,itime)
        caof(i,2,itime) = 0.0
        caof(i,3,itime) = 0.0
      end do
```

Step 3: **Assign maximum allowable total gas flow rate *ratmax* only if the current time step is the first time step *itime=1*. Initially, value of *ratmax* is set equal to proration factor *prorat_tech***

from input file *TECH.DAT* (see subprogram *CONVERT()*) where:

- ***prorat_tech<=0* means *ratmax* should be set to *aof*.**
- ***0<prorat_tech<1* means *ratmax* should be set to *prorat_tech*aof***
- ***prorat_tech>1* means *ratmax* should be set to *prorat_tech***

Note: *aof* is total gas flow rate based on absolute open flow.

```
      if (itime.eq.1) then
        if (ratmax.le.0) then
          ratmax = qtotal
        else if (ratmax.le.1.0) then
          ratmax = ratmax*qtotal
        end if
      end if
```

Step 4: **First, the reservoir is assumed to produce under minimum allowable wellhead pressure *premin* constraint (pressure specification) then calculate gas rates:**

- **Set wellhead pressure *pwh* to *premin*.**
- **Invoke subroutine *RATE1()* to calculate gas rates *qg* and total rate *qtotal* of primary well.**

```
      pwh = premin
      iwell = 1
      call rate1(pwh,qtotal,itime,ichg,iwell)
```

Step 5: **Change the well constraint to flow rate specification if *qtotal* (based on *premin*) is higher (or equal) to *ratmax* then calculate wellhead pressure:**

- **Invoke subroutine *RATE2()* to calculate wellhead pressure *pwh* with a constraint of *qtotal* equals to *ratmax*.**

```
      if (qtotal.ge.ratmax-1.)
   &    call rate2(pwh,qtotal,itime,ichg)
```

Step 6: **Subroutine *CALCPQ()* calculates wellhead and bottomhole pressures after rates have been determined.**

```
      call calcpq(pwh,qtotal,itime,ichg,maxtim)
      return
      end
```

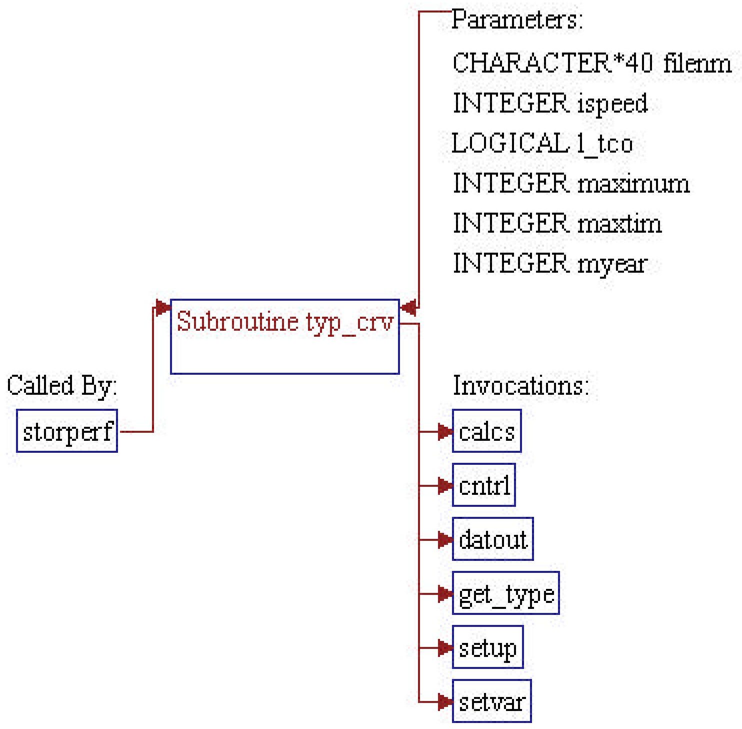
SUB-PROGRAM TYP_CRV()

MAIN THEME: This routine is the main driver of the Storage Reservoir Performance Module (SRPM) type curve routines. The current SRPM model solves only one development case (primary well case) and one pay grade (pay grade #2). The gathering pressure (wellhead pressure) is assumed to be the same for all wells in the field. The reservoir is allowed to produce against a minimum allowable wellhead pressure constraint as long as the total gas production rate does not exceed the maximum allowable total gas rate. Otherwise, the maximum allowable total gas rate constraint is utilized.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

filenm	Name of SRPM database file, GSAMID(4:14) (not currently used)
myear	Number of years of full production (years)
ispeed	Speedup option: 0=no speedup, 1=speedup
maximum	Maximum number of development cases (not currently used)
l_tco	Y/N flag for printing type curve output files (.TCO)

```
subroutine typ_crv(filenm,myear,ispeed,maximum,l_tco,maxtim)
```

Note: Include files, common block, and local variables.

```
include 'dimen.h'
include 'welldata.h'
include 'type_out.h'
include 'type111.h'
include 'type1.h'
include 'type2.h'
include 'type3.h'
include 'type4.h'
include 'type5.h'
include 'type6.h'
include 'type7.h'
include 'type8.h'
include 'type9.h'
include 'type10.h'
common /stchg/ iwin_yr
character*40 filenm
character*79 desc1$,desc2$
integer myear
logical l_tco
```

Step 2: **Run only for primary wells (development case = 1). Note that SRPM only considers the primary wells (no infills), therefore, parameter *maximum* (maximum number of development cases) is ignored.**

```
icase = 1
```

Note: Size of arrays of pressure functions (pseudo-pressure, viscosity, Z-factor, etc.) is set to 99.

```
narray = 99
```

Note: Absolute roughness of production/injection tubings is defaulted to 0.0006 inches

```
                absrns = .0006
```

Step 3: **Calculate number of time steps *maxtim***

Note: This is a storage mode (*i_prod_mode* <> 1):

- <u>*deltat*</u> is time step size (days)
- *prod_period* is the production period (days, should be less than 365 days).

```
                  maxtim = nint(prod_period/deltat)
```

Step 4: **Subroutine *SETVAR()* initializes time dependent variables to zero**

```
                call setvar(maxtim,narray,absrns,ichg)
```

Step 5: **Subroutine *SETUP()* sets up pressure function arrays**

Note: parameter *itype* is not currently used

```
                call setup(ispeed,itype)
```

Step 6: **Subroutine *CNTRL()* assigns time nodes (*time*), initializes pressure and rate variables, etc.**

```
                call cntrl(icase,maxtim,ispeed)
```

Step 7: **Flag to tell whether change has taken place (*ichg*) is set to zero (primary wells without stimulation).**

Note: this is done since the SRPM is only for primary wells without refrac option.

```
                ichg = 0
```

Note: *tchg* is used for infill or refrac option. Again, since these options are not used, here *tchg* is set -1 (for primary well).

```
                tchg = -1
```

Note: Original number of time steps is stored to variable *nmaxtim*

```
nmaxtim = maxtim
```

Note: Subroutine *CALCS()* controls type curve routines to generate rates and pressure profiles

```
do itime = 1,maxtim
  if (itime.le.nmaxtim) then
    call calcs(itime,icase,ichg,tchg,ispeed,nmaxtim)
  endif
enddo
```

Step 8: Subroutine *GET_TYPE()* assigns type curve output variables

```
call get_type(maxtim,icase,tchg)
```

Step 9: Subroutine *DATOUT()* prints out results to optional type curve output files (.TCO)

```
call datout(desc1$,desc2$,maxtim,icase,tchg,l_tco)
return
end
```

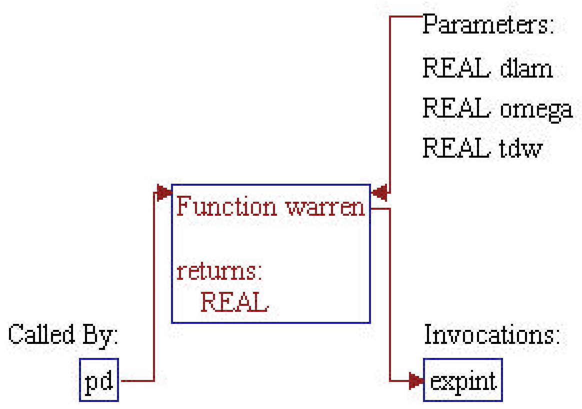
SUB-PROGRAM WARREN()

MAIN THEME: This routine computes the difference in dimensionless pressure for a conventional reservoir, compared to a naturally fractured reservoir, using the Warren and Root approach.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *tdw*	Dimensionless time based on wellbore radius
• *fcd*	Dimensionless fracture conductivity, kfw/kxf for vertically fractured and horizontal wells (modules 2 and 4)
• *shor*	Equivalent skin factor for a horizontal well
• *omega*	Warren and Root porosity-compressibility ratio: $omega = (phi\text{-}c)_{fractures} / (phi\text{-}c)_{total}$
• dlam	Warren and Root interporosity flow parameter: $dlam = 12(perm)_{matrix} / (perm)_{total} *$ (rw/frac spacing)**2

```
function warren(tdw,omega,dlam)
```

Step 2: **Dimensionless pressure difference (*Warren*) is calculated.**

```
if ((omega.le.0.).or.(omega.ge.1.).or.(dlam.le.0.)) then
  warren = 0.
else
  arg1 = dlam*tdw/omega/(1.-omega)
  arg2 = arg1*omega
  p1 = expint(arg1)
  p2 = expint(arg2)
  warren = 0.5*(-p1+p2)
end if
return
end
```

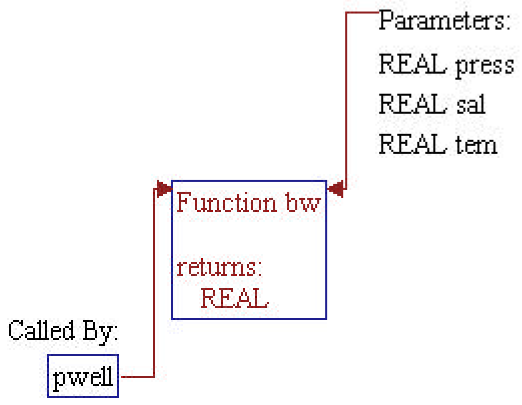
SUB-PROGRAM BW ()

MAIN THEME: Function to compute water formation volume factor, using a curve fit for the volume factor for water saturated with natural gas based on Dodson and Standing.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *press*	pressure, psia
• *tem*	temperature, degrees F
• *sal*	water salinity, ppm by weight

```
      function   bw (press,  tem,    sal)
```

Step 2: **Water formation volume factor is calculated.**

```
      bw = 0.991663 - 1.465e-6 * press + 5.984e-5 * tem +
     +       8.48e-7 * tem * tem
      return
      end
```

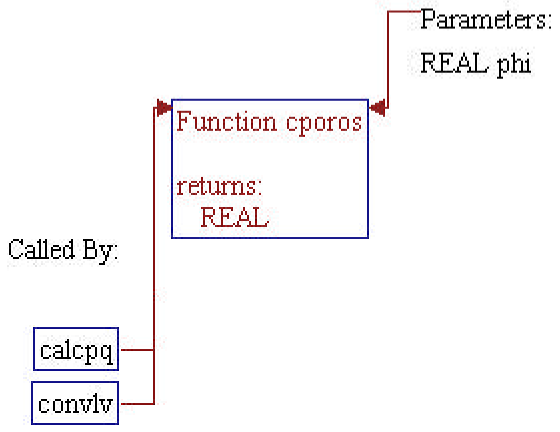
SUB-PROGRAM CPOROS ()

MAIN THEME: Function to compute pore volume compressibility using a curve fit to Hall's correlation.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameter of the subroutine:

- *phi* porosity, decimal

```
      function   cporos (phi)
```

Step 2: **Pore volume compressibility is calculated.**

```
      if (phi .lt. 0.02) then
              cporos =  0.191976 / phi * 1.0e-6
      else
              cporos = ((66.927 * phi + 20.195 ) * phi - 0.0735)
     +                 /(43.025 * phi + 1.) / phi * 1.0e-6
      end if
      return
      end
```

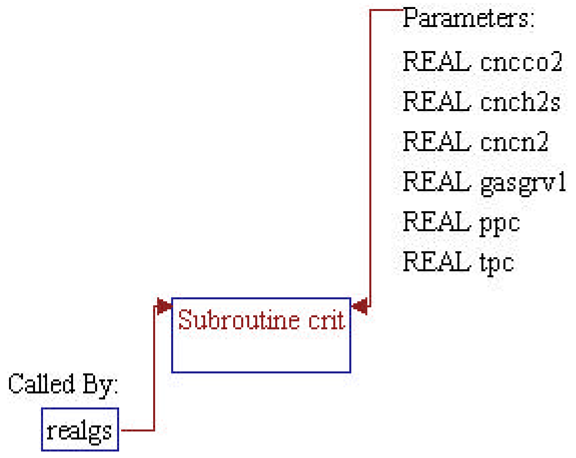
SUB-PROGRAM CRIT ()

MAIN THEME: Routine computes the pseudocritical properties of natural gases using standing's correlation. Corrections for nitrogen, hydrogen sulfide and carbon dioxide are determined by the method of Wichert and Aziz (1974).

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *gasgrv*	natural gas specific gravity (air=1)
• *cnch2s*	concentration hydrogen sulfide, mole fraction
• *cncco2*	concentration carbon dioxide, mole fraction
• *cncn2*	concentration nitrogen, mole fraction
• *tpc*	pseudo-critical temperature
• *ppc*	pseudo-critical pressure

```
      subroutine crit (gasgrv1, cnch2s, cncco2, cncn2,  tpc,    ppc)
```

Step 2: **Pseudo-critical properties are calculated.**

Note: first compute properties of hydrocarbon fraction

```
      cnchc = (1. - cncn2 - cncco2 - cnch2s)
      ghc = (gasgrv1 - 0.967*cncn2 - 1.52*cncco2 - 1.18*cnch2s)/cnchc
```

Note: check if gravity of hydrocarbon fraction is at least methane

```
      if (ghc .lt. 0.554) ghc = 0.554
```

Note: compute the pseudocriticals of the hydrocarbon fraction

```
      ppchc = 677. + 15.0 * ghc - 37.5 * ghc * ghc
      tpchc = 168. + 325. * ghc - 12.5 * ghc * ghc
```

Note: compute the pseudocriticals of the entire mixture.

```
      ppcm = ppchc*cnchc + 493.*cncn2 + 1071.*cncco2 + 1306.*cnch2s
      tpcm = tpchc*cnchc + 227.*cncn2 +  548.*cncco2 +  672.*cnch2s
```

Note: now adjust for impurities with wichert and aziz correction.

```
      eps  = 120.*((cnch2s+cncco2)**0.9 - (cnch2s+cncco2)**1.6) +
   +       15.*(cnch2s**0.5-cnch2s**4.0)
      tpc = tpcm - eps
      ppc = ppcm * tpc / (tpc + cnch2s * (1. - cnch2s) * eps)
      return
      end
```

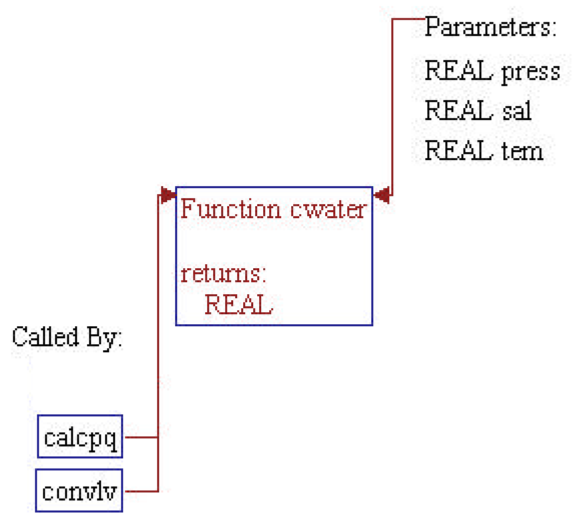
SUB-PROGRAM CWATER ()

MAIN THEME: Function to compute water compressibility. Osif's correlation (SPE Reservoir Engineering, Feb., 1988, pp. 175-181) is used.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *press*	pressure, psia
• *tem*	temperature, degrees f
• *sal*	water salinity, ppm by weight

```
      function   cwater (press,  tem,    sal)
```

Step 2: **Water compressibility is calculated.**

Note: first, convert water salinity from ppm to g/l, using an adaptation of a correlation for water specific volume presented by Rowe and Chou, Jour. Ch. and Eng. data, v. 15, no. 1, pp. 61-66. Use 3 iterations to find specific volume.

```
      t = (tem - 32.0) / 1.8 + 273.
      at = 5.916365 - 0.01035794 * t + 0.9270048e-5 * t * t
     +     - 1127.522 / t + 100674.1 / (t * t)
      dt = -2.5166 + 0.0111766 * t - 0.170552e-4 * t * t
      et = 2.84851 - 0.0154305 * t + 0.223982e-4 * t * t
      x = sal / 1.e6
      do iter = 1, 3
              v = at + dt * x + et * x * x
              x = sal / 1.e6 / v
      end do
```

Note: now apply Osif's correlation.

```
      conc = x * 1000.
      f = 7.033 * press + 54.15 * conc - 537 * tem + 403300.
      cwater = 1. / f
      return
      end
```

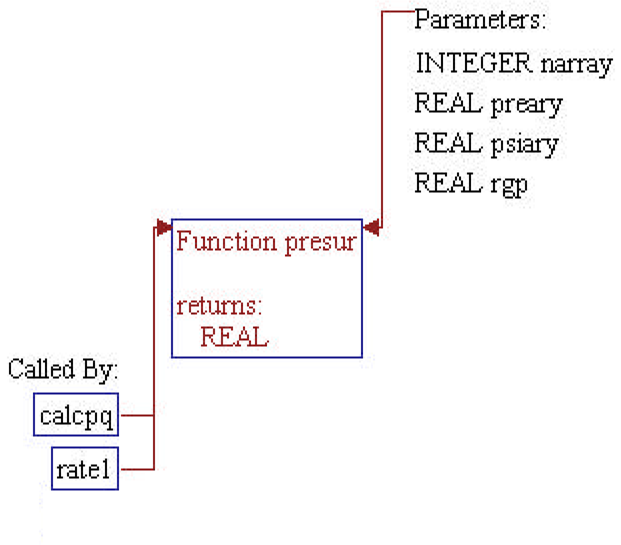
SUB-PROGRAM PRESUR ()

MAIN THEME: Function for inverse table look up of pressure for given real gas potential, interpolating on p**2. for rgp <= 0., return presur = 0.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: Subroutine declarations and definitions.

Note: Parameters of the subroutine:

• *Rgp*	real gas potential, psia**2/cp
• *narray*	number of points in table
• *preary*	array of pressure data, psia
• *psiary*	array of real gas potential data, psia**2/cp

```
      function   presur (rgp,    narray, preary, psiary)
```

Note: Local variables.

```
      dimension preary(99),psiary(99)
```

Step 2: Pressure as a function of pseudo-pressure is determined.

```
      if (rgp .le. 0.) then
              presur = 0.
      else if (rgp .ge. psiary(narray)) then
              presur = preary(narray)
      else if (rgp .le. psiary(1)) then
              presur = sqrt( rgp / psiary(1)) * preary(1)
      else
              i0 = 1
              i1 = narray
```

Note: find correct index in table lookup by bisection

```
              dn = float(narray)
              d  = log(dn)/log(2.) + 1.
              imax = int(d)
              do i = 1, imax
                      if (i1 .gt. i0 + 1) then
                              i2 = (i0 + i1 ) / 2
                              if (rgp .le. psiary(i2)) then
                                      i1 = i2
                              else
                                      i0 = i2
                              end if
                      end if
              end do
              presur = sqrt((rgp-psiary(i0))/(psiary(i1)-psiary(i0))*
     +                (preary(i1)**2-preary(i0)**2)+preary(i0)**2)
      end if
      return
      end
```

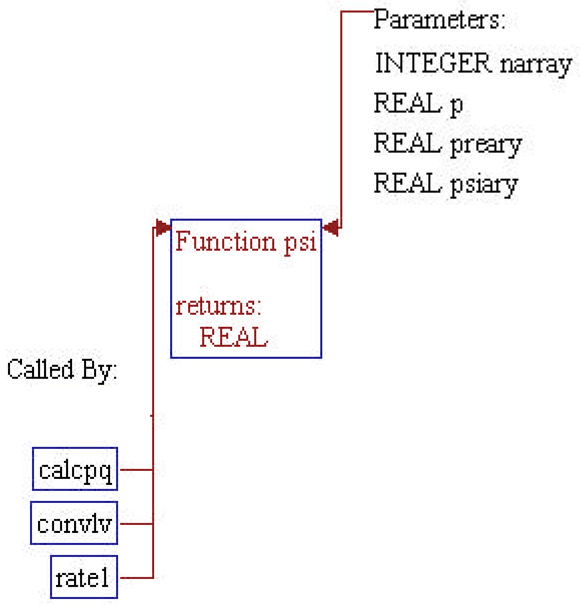
SUB-PROGRAM PSI ()

MAIN THEME: Function for table look up of real gas potential interpolating on p**2. for p <= 0., return psi = 0.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *P*	pressure, psia
• *narray*	number of points in table
• *preary*	array of pressure data, psia
• *psiary*	array of real gas potential data, psia**2/cp

```
        function   psi    (p,narray, preary, psiary)
```

Note: Local variables.

```
        dimension preary(99),psiary(99)
```

Step 2: **Psudo-pressure as a function of pressure is determined.**

```
        if (p .le. 0.) then
                psi = 0.
        else if (p .ge. preary(narray)) then
                psi = psiary(narray)
        else
                x = p / preary(1)
                ix = int (x)
                if (ix .gt. narray - 1) ix = narray – 1
                if (x .ge. 1.) then
                        psi = psiary(ix) + (psiary(ix+1) - psiary(ix)) *
     +                      ( p**2 - preary(ix)**2) / (preary(ix+1)**2 -
     +                        preary(ix)**2)
                else
                        psi = psiary(1) * x**2
                end if
        end if
        return
        end
```

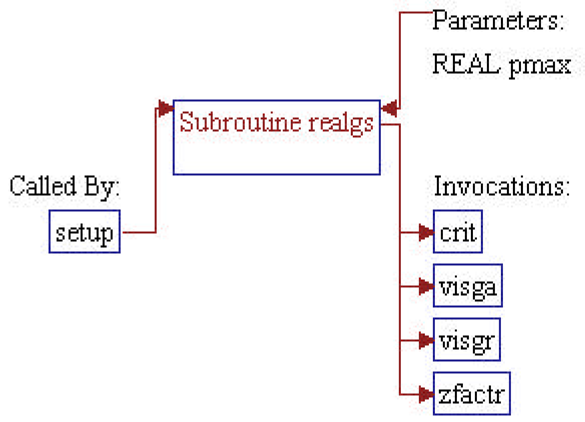
SUB-PROGRAM REALGS()

MAIN THEME: Routine computes real gas potential and sets up a table at *narray* pressure increments from 0 to *pinit*. Pseudocritical properties are determined using the routine CRIT(). Z-factors are determined using the function ZFACTR() and gas viscosity is determined using the functions VISGA() and VISGR(). Simpson's rule is used to numerically integrate the real gas potential.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

- *pmax* maximum pressure for table, psia

```
                subroutine realgs (pmax)
```

Note: Include files.

```
                 include 'type1.h'
                 include 'type2.h'
```

Note: determine critical properties and viscosity at 1 atmosphere

```
                call crit(gasgrv1, cnch2s, cncco2, cncn2, tpc, ppc)
                tpr = (tem + 460.) / tpc
                va  = visga (gasgrv1, tem, cnch2s, cncco2, cncn2)
```

Note: initialize variables and arrays

```
                do i = 1, narray
                        preary(i) = float(i) * pmax / float(narray)
                        psiary(i) = 0.
                end do
                p1   = 0.0
                v1   = 1.0
                z1   = 1.0
                ppsi = 0.0
                f1   = 0.0
                dp   = preary(1)
```

Note: perform numerical integration using Simpson's rule with one intermediate pressure point between each tabulated point.

```
                do i = 1, narray
                        p2  = (preary(i) + p1) / 2.
                        ppr = p2 / ppc
                        v2  = visgr (tpr, ppr)
                        z2  = zfactr(tpr, ppr)
                        f2  = 2. * p2 / (v2 * va * z2)
                        p3  = preary(i)
                        ppr = p3 / ppc
                        v3  = visgr (tpr, ppr)
                        z3  = zfactr(tpr, ppr)
                        f3  = 2. * p3 / (v3 * va * z3)
                        ppsi = ppsi + (f1 + 4. * f2 + f3) / 6. * dp
                        psiary(i) = ppsi
                        visary(i) = v3 * va
                        zary(i)   = z3
                        p1 = p3
                        f1 = f3
                end do
```

```
      return
      end
```

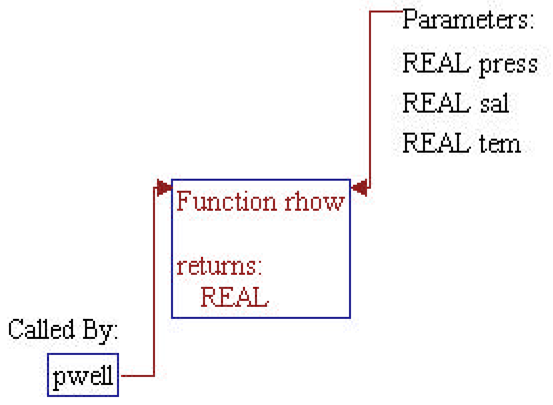
SUB-PROGRAM RHOW()

MAIN THEME: Function to compute water density, using a curve fit to the correlation in the Petroleum Engineers Handbook.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *press*	pressure, psia
• *tem*	temperature, degrees F
• *sal*	water salinity, ppm by weight

```
      function   rhow   (press,  tem,    sal)
```

Step 2: **Water density is calculated.**

```
      x = sal / 120000.
      rho1 = (1.001125 + 0.095062*x + 0.001688*x*x) +
     +       (1.25     - 20.0    *x + 3.75    *x*x) *tem    *1.e-5+
     +       (-10.15625+ 5.859375*x -1.171875 *x*x) *tem**2 *1.e-7
      drho = 0.0226 * press / 6000. / rho1 ** 1.3
      rhow = rho1 + drho
      return
      end
```

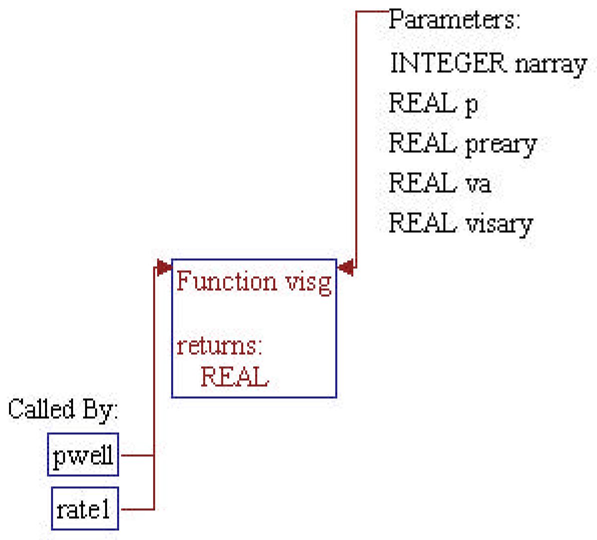
SUB-PROGRAM VISG()

MAIN THEME: Function for table look up of gas viscosity using linear interpolation.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: Subroutine declarations and definitions.

Note: Parameters of the subroutine:

• *p*	pressure, psia
• *narray*	number of points in table
• *preary*	array of pressure data, psia
• *visary*	array of gas viscosity data, centipoise
• *va*	gas viscosity at one atmosphere, centipoise

```
      function   visg (p,narray, preary, visary, va)
```

Note: Parameters of the subroutine:

```
      dimension preary(99),visary(99)
```

Step 2: Gas viscosity is calculated.

```
      if (p .le. 0.) then
              visg = va
      else if (p .ge. preary(narray)) then
              visg = visary(narray)
      else
              x = p / preary(1)
              ix = int (x)
              if (ix .gt. narray - 1) ix = narray - 1
              if (x .ge. 1.) then
                      visg = visary(ix) + (visary(ix+1)-visary(ix)) *
   +                       ( x – preary(ix)/preary(1))
              else
                      visg = va + (visary(1) - va) * x
              end if
      end if
      return
      end
```

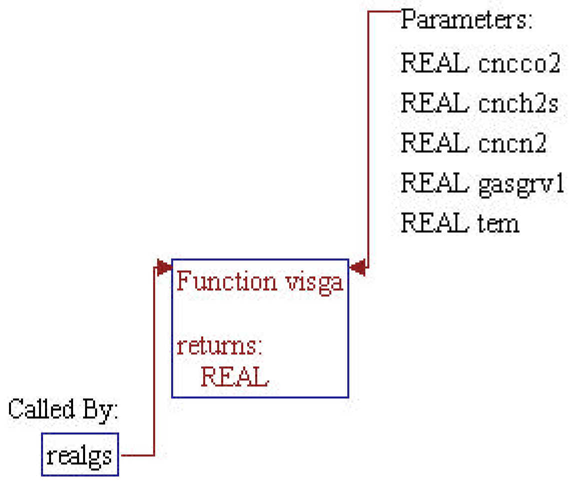
SUB-PROGRAM VISGA()

MAIN THEME: This function determines natural gas viscosity at a pressure of 1 atm, corrected for hydrogen sulfide, carbon dioxide and nitrogen. The function was adapted from a program presented in the ERCB manual.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *gasgrv*	natural gas specific gravity (air=1)
• *tem*	gas temperature, degrees f
• *cnch2s*	concentration hydrogen sulfide, mole fraction
• *cncco2*	concentration carbon dioxide, mole fraction
• *cncn2*	concentration nitrogen, mole fraction

```
      function   visga  (gasgrv1, tem, cnch2s, cncco2, cncn2)
```

Step 2: **Gas viscosity at 1 atm is calculated.**

Note: check allowable ranges on gas gravity and temperature.

```
      g = min(gasgrv1,1.5)
      g = max(g,0.5)
      t = min(tem,400.)
      t = max(t,40.)
```

Note: compute the uncorrected viscosity, *visgu*.

```
      visgu  = 0.0126585 - 0.611823e-02 * g + 0.164574e-02 *
     +          g * g + 0.164574e-04 * t - 0.719221e-06 *
     +          g * t - 0.609046e-06 * g * g * t
```

Note: correct for h2s, co2 & n2.

```
      c = min(cnch2s,0.15)
      c = max(c,0.)
      corh2s = (0.000113 * c/100. * g - 0.000038 * c/100. +
     +          0.000001) * (1.0/(1.0 + g)) + 0.000001
      c = min(cncco2,0.15)
      c = max(c,0.)
      corco2 = (0.000134 * c/100. * g - 0.000004 * c/100. +
     +          0.000004 * g) * (1.0/(1.0 + g)) - 0.000003
      c = min(cncn2,0.15)
      c = max(c,0.)
      corn2  = (0.000170 * c/100. * g - 0.000021 * c/100. +
     +          0.000010 * g)* (1.0/(1.0 + g)) - 0.000006
```

Note: now calculate the viscosity at 1 atm pressure, *visga*.

```
      visga  = visgu + corh2s + corco2 + corn2
      return
      end
```

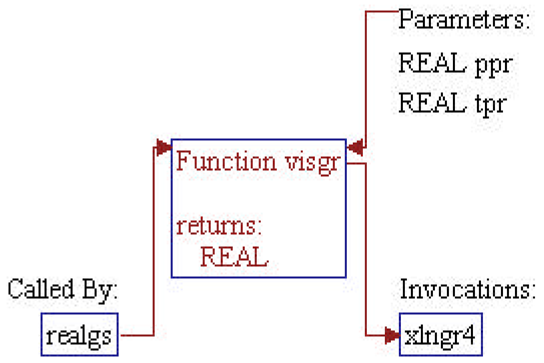
SUB-PROGRAM VISGR()

MAIN THEME: This function determines the reduced viscosity of natual gas, adapted from a program presented in the ERCB manual. The reduced viscosity is the gas viscosity at a given pressure and temperature, divided by the gas viscosity at one atmosphere pressure and that temperature.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

- *tpr* pseudo-reduced temperature
- *ppr* pseudo-reduced pressure

```
      function   visgr (tpr, ppr)
```

Note: Local variables and data.

```
      dimension temtbl(13), prstbl(22), vistbl(22,13)
      data temtbl /1.05,1.10,1.15,1.20,1.30,1.40,1.50,1.60,1.75,2.00,
     +      2.25,2.50,3.00/
      data prstbl /0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0,1.2,1.4,
     +      1.6,1.8,2.0,3.0,4.0,6.0,8.0,10.0,15.0,20.0/
      data ((vistbl(i,j),i=1,22),j=1,7)/
     +1.000,1.012,1.025,1.050,1.075,1.100,1.145,1.195,1.285,1.400,1.760,
     +2.285,2.865,3.290,3.800,4.760,5.500,6.500,7.250,7.900,9.080,9.850,
     +1.000,1.011,1.023,1.043,1.065,1.086,1.120,1.150,1.195,1.255,1.435,
     +1.700,2.070,2.465,2.800,3.850,4.655,5.720,6.500,7.100,8.260,9.000,
     +1.000,1.010,1.021,1.036,1.055,1.073,1.095,1.120,1.145,1.175,1.280,
     +1.420,1.590,1.850,2.100,3.225,3.975,5.030,5.820,6.385,7.550,8.250,
     +1.000,1.009,1.019,1.030,1.045,1.060,1.070,1.085,1.110,1.135,1.195,
     +1.285,1.425,1.570,1.750,2.600,3.350,4.380,5.200,5.740,6.900,7.600,
     +1.000,1.008,1.017,1.027,1.040,1.054,1.063,1.075,1.100,1.115,1.155,
     +1.215,1.285,1.360,1.450,2.020,2.560,3.490,4.185,4.700,5.790,6.500,
     +1.000,1.007,1.015,1.024,1.035,1.048,1.056,1.067,1.089,1.100,1.135,
     +1.185,1.235,1.280,1.335,1.680,2.100,2.790,3.380,3.860,4.790,5.410,
     +1.000,1.006,1.013,1.021,1.030,1.042,1.049,1.059,1.078,1.087,1.120,
     +1.150,1.185,1.220,1.260,1.500,1.785,2.325,2.820,3.230,4.060,4.610/
      data ((vistbl(i,j),i=1,22),j=8,13)/
     +1.000,1.005,1.011,1.018,1.025,1.036,1.042,1.051,1.067,1.075,1.108,
     +1.134,1.160,1.180,1.215,1.385,1.595,2.020,2.425,2.790,3.490,4.025,
     +1.000,1.004,1.009,1.015,1.021,1.030,1.035,1.043,1.056,1.065,1.090,
     +1.110,1.125,1.145,1.165,1.295,1.435,1.780,2.070,2.375,2.990,3.490,
     +1.000,1.003,1.007,1.012,1.017,1.024,1.028,1.035,1.045,1.050,1.060,
     +1.070,1.080,1.095,1.110,1.200,1.290,1.500,1.710,1.950,2.460,2.875,
     +1.000,1.002,1.005,1.009,1.013,1.018,1.021,1.027,1.034,1.037,1.045,
     +1.055,1.065,1.075,1.085,1.145,1.210,1.340,1.485,1.665,2.085,2.460,
     +1.000,1.001,1.003,1.006,1.009,1.012,1.015,1.019,1.023,1.025,1.030,
     +1.040,1.050,1.060,1.065,1.110,1.155,1.245,1.355,1.485,1.830,2.150,
     +1.000,1.000,1.001,1.003,1.005,1.007,1.009,1.011,1.013,1.015,1.020,
     +1.025,1.030,1.035,1.040,1.060,1.095,1.140,1.190,1.265,1.495,1.730/
```

Step 2: **Reduced viscosity is calculated.**

Note: check upper and lower limits of input parameters. If value is outside the range, set *visgr=1* and return.

```
      if (tpr.lt.1.02 .or. tpr.gt.3.01 .or. ppr.lt.0.01) then
              visgr  = 1.00
              return
      end if
```

Note: locate *tpr*, *ppr* in the indices

```
      j = 12
      do j1 = 11, 3, -1
              if (temtbl(j1) .ge. tpr) j=j1
      end do
      i = 22
      do i1 = 21, 2, -1
              if (prstbl(i1) .ge. ppr) i=i1
      end do
```

Note: use XLNGR4(), a four-point Lagrange interpolation routine, to interpolate on temperature. Use linear interpolation on *1/(pr+1)* to interpolate on pressure.

```
      call xlngr4 (tpr, temtbl(j-2), temtbl(j-1), temtbl(j),
   +          temtbl(j+1), visj, vistbl(i,j-2), vistbl(i,j-1),
   +          vistbl(i,j), vistbl(i,j+1))
      call xlngr4 (tpr, temtbl(j-2), temtbl(j-1), temtbl(j),
   +          temtbl(j+1), visi, vistbl(i-1,j-2), vistbl(i-1,j-1),
   +          vistbl(i-1,j), vistbl(i-1,j+1))
      visgr = visi + ((1./(1.+ppr) - 1./(1.+prstbl(i-1)))/
   +          (1./(1.+prstbl(i)) - 1./(1.+prstbl(i-1))))*(visj-visi)
      return
      end
```

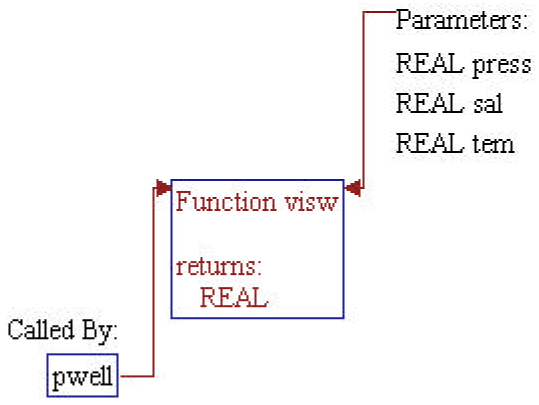
SUB-PROGRAM VISW()

MAIN THEME: Function to compute water viscosity, using Meehan's correlation.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *press*	pressure, psia
• *tem*	temperature, degrees F
• *sal*	water salinity, ppm by weight

```
      function   visw (press,  tem,    sal)
```

Step 2: **Water viscosity is calculated.**

```
      conc = sal * 0.0001
      sqconc = sqrt(conc)
      sc2 = 1. + 0.00187 * sqconc + 0.000218 * sqconc * conc * conc +
     +    ( sqrt(tem) - 0.0135 * tem) * (0.00276 - 0.000344 * sqconc)*
     +      conc
      sp = 1. + 3.5e-12 * press * press * (tem - 40.)
      visw  = sc2 * sp * 0.02414 * 10. ** (446.04 / (tem + 208.))
      return
      end
```

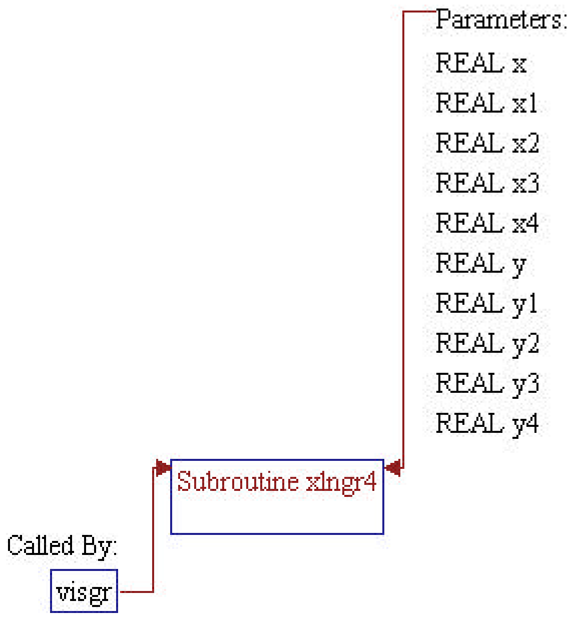
SUB-PROGRAM XLNGR4()

MAIN THEME: This subroutine performs a four point lagrange interpolation for the function VISGR() to interpolate the viscosity ratio based on pseudo-reduced temperature. The routine was adapted from the ERCB manual. The general lagrange equation (K.L. Neilson, methods in numerical analysis, the Macmillan Company, 1956) is solved for a y value corresponding to a given x lying in the range of four points: (x1,y1), (x2,y2), (x3,y3) and (x4,y4).

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: Subroutine declarations and definitions.

Note: Parameters of the subroutine:

• *x*	x-coordinate of desired point
• *xi*	x-coordinate of tabulated data, i=1,2,3 or 4
• *yi*	y-coordinate of tabulated data, i=1,2,3 or 4

```
                subroutine xlngr4 (x, x1, x2, x3, x4, y, y1, y2, y3, y4)
```

Step 2: 4-point Lagrange interpolation is performed.

```
                     a1 = x1 - x2
                     a2 = x1 - x3
                     a3 = x1 - x4
                     a4 = x2 - x3
                     a5 = x2 - x4
                     a6 = x3 - x4
                     b1 = x  - x1
                     b2 = x  - x2
                     b3 = x  - x3
                     b4 = x  - x4
                y  = b2 / a1 * b3 / a2 * b4 / a3 * y1 -
             +       b1 / a1 * b3 / a4 * b4 / a5 * y2 +
             +       b1 / a2 * b2 / a4 * b4 / a6 * y3 -
             +       b1 / a3 * b2 / a5 * b3 / a6 * y4
                return
                end
```

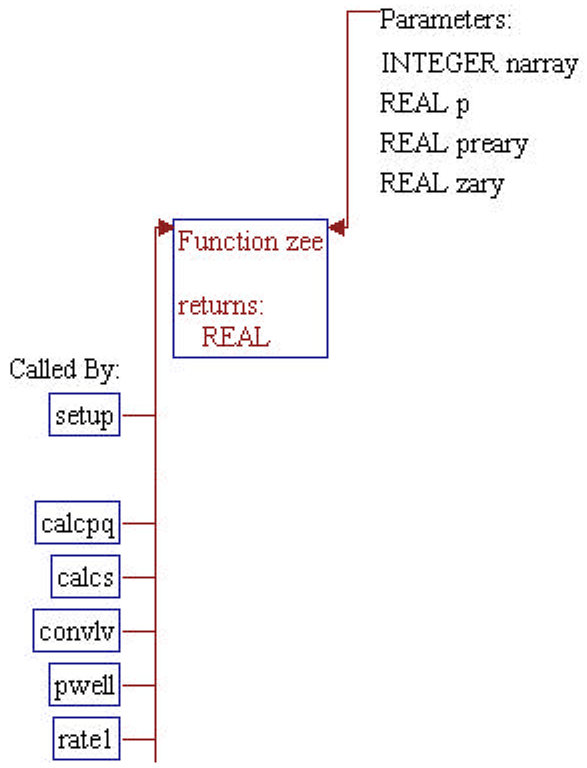
SUB-PROGRAM ZEE()

MAIN THEME: Function for table look up of Z-factor using linear interpolation. for p <= 0., return z = 1.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: Subroutine declarations and definitions.

Note: Parameters of the subroutine:

• *p*	pressure, psia
• *narray*	number of points in table
• *preary*	array of pressure data, psia
• *zary*	array of z factor data, dimensionless

```
      function   zee (p, narray, preary, zary)
```

Note: Local variables.

```
      dimension preary(99),zary(99)
```

Step 2: Linear interpolation for gas Z-factor is performed.

```
      if (p .le. 0.) then
              zee = 1.
      else if (p .ge. preary(narray)) then
              zee = zary(narray)
      else
              x = p / preary(1)
              ix = int (x)
              if (ix .gt. narray - 1) ix = narray - 1
              if (x .ge. 1.) then
                      zee = zary(ix) + (zary(ix+1) - zary(ix)) *
     +                    ( x - preary(ix)/preary(1))
              else
                      zee = 1. + (zary(1) - 1.) * x
              end if
      end if
      return
      end
```

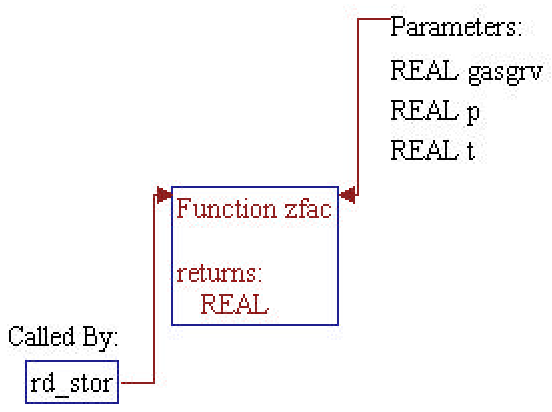
SUB-PROGRAM ZFAC()

MAIN THEME: Correlation to calculate z-factor (compressibility factor) using Dranchuk-Abou-Kassem correlation (1975).

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: Subroutine declarations and definitions.

Note: Parameters of the subroutine:

- *gasgrv* gas gravity
- *p* pressure, psia
- *t* temperature, degree F

```
      function zfac(gasgrv,p,t)
```

Note: Data.

```
      data a1,a2,a3,a4,a5 /0.3265,-1.070,-0.5339,0.01569,-0.05165/
      data a6,a7 /0.5475,-0.7361/
      data a8,a9,a10,a11 /0.1844,0.1056,0.6134,0.7210/
```

Step 2: Gas Z-factor is calculated.

```
      iter=0
      ppc = 756.8 - 131.0*gasgrv - 3.6*gasgrv**2
      tpc = 169.2 + 349.5*gasgrv - 74.0*gasgrv**2
      pr = p/ppc
      tr = (t + 460.0)/tpc
      j=1
      dr=1.0
      tr2=tr**2
      tr3=tr**3
      tr4=tr**4
      c1=a7+a8/tr
      c0=a1*tr+a2+a3/tr2+a4/tr3+a5/tr4
      c2=a6*tr+c1
      c3=-c1*a9
      c4=a10/tr2
      if (pr-30.0) 10,10,200
10    if (tr-1.0) 20,20,30
20    j=0
      dr=0.0
      deldr=0.1
30    if (tr-3.0) 40,40,200
40    do 160 iter=1,100
      if (j) 50,50,60
50    dr1=dr
      dr=dr+deldr
60    dr2=dr**2
      dr5=dr**5
      t1=c0*dr
      t2=c2*dr2
      t3=c3*dr5
      t4=c4*dr2
      t5=a11*dr2
      t6=exp(-t5)
      pn=(tr+t1+t2+t3)*dr+t4*dr*(1.0+t5)*t6
      dp=tr+2.0*t1+3.0*t2+6.0*t3+t4*t6*(3.0+3.0*t5-2.0*t5*t5)
      if (j) 70,70,100
70    prcal=pn/0.27
      if (abs(prcal-pr)-0.001) 170,170,80
80    if (prcal-pr) 160,170,90
90    dr=dr1
      deldr=deldr/2.0
      go to 160
```

```
100     dr1=dr-(pn-0.270*pr)/dp
        if (dr1) 110,110,120
110     dr1=0.5*dr
120     if (dr1-2.2) 140,140,130
130     dr1=dr+0.9*(2.2-dr)
140     if (abs(dr-dr1)-0.00001) 170,150,150
150     dr=dr1
160     continue
170     zfac=0.270*pr/(dr*tr)
200     return
        end
```

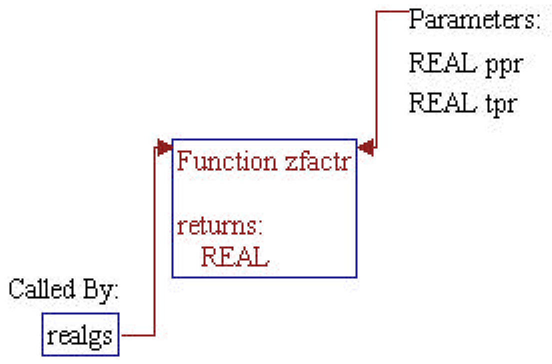
SUB-PROGRAM ZFACTR()

MAIN THEME: This function computes the gas deviation factor z as a function of pressure and temperature. The function uses the Hall and Yarborough correlation to reproduce and extend the Standing-Katz Z-actor charts. A constrained Nwton-Raphson procedure is used to solve the equation of state.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

- *tpr* pseudo-reduced temperature
- *ppr* pseudo-reduced pressure

```
      function   zfactr (tpr, ppr)
```

Step 2: **Gas Z-factor is calculated.**

Note: check ranges of *ppr, tpr*. if outside ranges, then *z = 1*.

```
      if ((tpr.lt.0.9999) .or. (ppr.le.0.001)) then
              zfactr = 1.0
              return
      end if
```

Note: use up to 10 Newton- Raphson iterations.

```
      a = 0.06125 / tpr  * exp (-1.2 * (1.0 - 1.0 / tpr)**2 )
      b = 14.76   / tpr  - 9.76  / tpr**2 + 4.58 / tpr**3
      c = 90.7    / tpr  - 242.2 / tpr**2 + 42.4 / tpr**3
      d = 2.82    / tpr  + 2.18
      y  = 0.001
      f  = 1.
      dy = 1.
      do iz = 1, 10
              if (abs(f) .gt. 0.00001) then
                      f  = - a*ppr + ( y + y**2 + y**3 - y**4 )
     +                       / (1. - y)**3 - b*y*y + c*y**d
                      fp = ( 1. + 4.*y + 4.*y**2 - 4.*y**3 +y**4 )
     +                       / (1. - y)**4 - 2*b*y + c*d*y**(d-1)
                      dy = f/fp
                      y  = y - dy
                      if (y .gt. 0.6) y = 0.6
                      if (y .lt. 0.1e-5) y = 0.1e-5
              end if
      end do
      zfactr = a * ppr / y
      return
      end
```

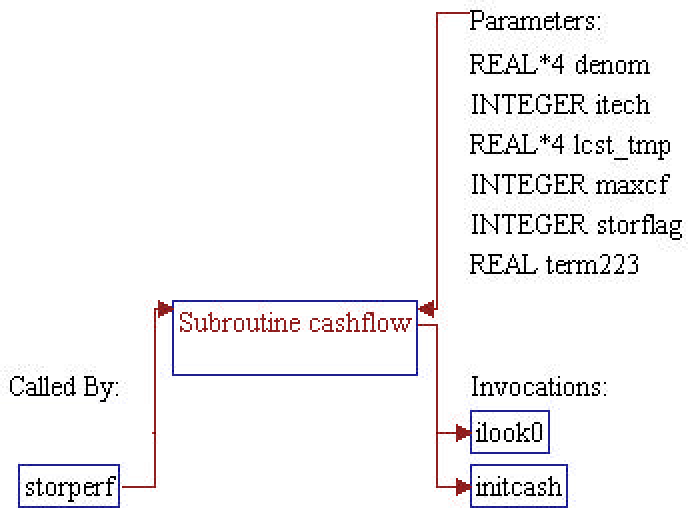
SUB-PROGRAM CASHFLOW()

MAIN THEME: This routine performs a discounted cash flow analysis for every gas reservoir (i.e. performs a pro-forma cash flow analysis for every reservoir processed)

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameter of the subroutine:

• *itech*	Technology flag (should be 1 for current technology only)
• *maxcf*	Flag for environmental RP run (currently it is set to “0” for nonenvironmental run).
• *storflag*	Storage flag (should be 1 for primary well and current technology only)
• *lcst_tmp*	Levelized investment cost
• *denom*	NPV of working gas

```
      subroutine cashflow(itech,maxcf,storflag,lcst_tmp,denom)
```

Note: Include files and local variables.

```
      include 'dimen.h'
      include 'global.h'
      include 'cashflow.h'
      include 'costing.h'
      include 'cost.h'
      include 'field.h'
      include 'tax_nat.h'
      include 'tax_reg.h'
      include 'unitcost.h'
      include 'gsamvar.h'
      include 'storlp.h'
      real*4 schedule(8)
      integer iyr,iyr1,itech
      real*4 temp
      real*4 tgglcd(qyr)
      integer storflag
      real*4  numer,denom,lcst_tmp
```

Step 2: **Depreciation schedule data is assigned. (MACRS Schedule)**

Note: “nyr” is the number of years for the potential storage run.

```
      data schedule/0.1423,0.2449,0.1749,0.1249,
     &    0.0893,0.0893,0.0893,0.0446/
      nyr = nyrset_storage
```

Step 3: **Sub-program INITCASH is invoked to initialize cash flow variables.**

Note: The cash flow variables are declared in header file CASHFLOW.H. “numer” is the summation of all costs for levelized investment calculation.

```
          call initcash
          numer = 0.0
          denom = 0.0
```

Step 4: **Sub-program ILOOK0() is invoked to search for State ID given in variable *state* in state tax array *tax_st()* and stores the pointer to variable *istate.***

Note: If no match is found (*istate=0*), the default location/pointer (*qstate+1*) is utilized.

```
          call ilook0(state,tax_st,ntax_st,istate)
          if(istate.eq.0) istate=qstate+1
```

Step 5: **Array of total G&G lease cost depletion (*tgglcd()*) is initialized to zero.**

```
          do iabb=1,qyr
            tgglcd(iabb)=0.0
          enddo
```

Step 6: **Loop of years for cash flow calculation is initialized.**

```
          do iyr=1,nyr
```

Step 7: **Tangible development and exploratory well costs (*tang_dwc()* and *tang_ewc()*) and intangible development and exploratory well costs (*intang_dwc()* and *intang_ewc()*) in each year are calculated. Variables tang_m, intang_m and oam_m are tangible cost, intangible cost and O&M cost multipliers. These are calculated in program UNITCOST.FOR as a function of gas price.**

```
         tang_dwc(iyr)=(dwc(iyr)+fix(iyr))*dwc_tan(itech)*tang_m(iyr)
         tang_ewc(iyr)=ewc(iyr)*ewc_tan(itech)*tang_m(iyr)
         intang_dwc(iyr)=dwc(iyr)*(1-dwc_tan(itech))*intang_m(iyr)
         intang_ewc(iyr)=ewc(iyr)*(1-ewc_tan(itech))*intang_m(iyr)
```

Step 8: **Adjusted gross sales (*adjgross()*), net sales (*netsales()*), G&A on expensed items (*ga_exp()*), and intangible investment (*ii()*) in each year are calculated.**

```
         adjgross(iyr)=
      &   mwgc*gprice(1,iyr)/1000.0-gravpen(iyr)-transcst(iyr)
         netsales(iyr)=adjgross(iyr)-adjgross(iyr)*royrate
         ga_exp(iyr)=ga_exp_m(itech)*(inj(iyr)+oam(iyr)+eoam(iyr))
```

```
           ii(iyr)=intang_ewc(iyr)+intang_dwc(iyr)+icap(iyr)+eicap(iyr)
        &      + stor_gas_cost(iyr)
```

Step 9: **Intangible capitalized (*intcap()*) in each year is calculated.**

Note: First, *intcap()* for drilling cost is calculated if it is requested in input file TAX_NAT.DAT (*cidc=.true.*). The *intcap()* is then modified if environmental and/or other intangibles are also requested to be capitalized (*ce=.true.* and/or *coi=.true.*) in input file TAX_NAT.DAT.

```
              if(cidc) intcap(iyr)=
           &    intcap(iyr)+piic*intang_dwc(iyr)+piic*intang_ewc(iyr)
              if(ce) intcap(iyr)=intcap(iyr)+piic*eicap(iyr)
              if(coi) intcap(iyr)=intcap(iyr)+piic*icap(iyr)
```

Step 10: **Tangible investment (*ti()*) and total capitalized investment (*tci()*) are calculated.**

```
              ti(iyr)=etcap(iyr)+tang_dwc(iyr)+tang_ewc(iyr)+otc(iyr)
              tci(iyr)=ti(iyr)+intcap(iyr)
```

Step 11: **Total capitalized investment adjustment (*tciadj()*) is calculated.**

Note: Logical variables *eortc* (allow enhanced oil recovery tax credit always set to “no”), *tcoti* (allow tax credit on tangible investments), *tdtc* (allow tangible development tax credit), *eec* (include expense environmental costs), and *ettc* (allow environmental tangible tax credit) control the *tciadj()* calculation. Except for *eortc*, YES/NO responses for all of these logical variables are obtained from input file TAX_NAT.DAT.

```
              if(eortc) tciadj(iyr)=tciadj(iyr)+eortcr*tci(iyr)
              if(tcoti) then
                if(yr1.ge.iyr) tciadj(iyr)=tciadj(iyr)+
           &    tdtcr*(tang_dwc(iyr)+tang_ewc(iyr))+ ettcr*etcap(iyr)
              else
                if(tdtc) tciadj(iyr)=tciadj(iyr)+
           &    tdtcr*(tang_dwc(iyr)+tang_ewc(iyr))
                if(.not. eec .and. ettc) tciadj(iyr)=tciadj(iyr)+
           &    ettcr*etcap(iyr)
              endif
```

Step 12: **Total operating cost (*toc()*) and depreciation (*depr()*) are calculated.**

Note: Depreciation is calculated only if depreciable/capitalize base (*cap_base()*) is greater than zero.

```
        cap_base(iyr)=tci(iyr)-tciadj(iyr)
        ga_cap(iyr)=ga_cap_m(itech)*(ii(iyr)+ti(iyr))
        toc(iyr)=inj(iyr)+oam(iyr)+eoam(iyr)+ga_exp(iyr)+
     &  ga_cap(iyr)+stim(iyr)
     & + gasinje(iyr) *(gprice(1,iyr) )
        if(cap_base(iyr).gt.0) then
          do iyr1=0,7
            if(iyr+iyr1 .le. qyr)
     1      depr(iyr+iyr1)=depr(iyr+iyr1)+
     2      cap_base(iyr)*schedule(iyr1+1)
          enddo  !iyr1
        endif
```

Step 13: **Expensed G&G and lease acquisition (*aggla()*) is calculated.**

```
        eggla(iyr)=la(iyr)*(1-plac) + gg(iyr)*(1-pggc)
```

Step 14: **Severance tax (*sevtax()*) is calculated.**

Note: If forgiveness of state taxes is allowed (*fsttax=.true.*, specified in input file TAX_NAT.DAT) and within the eligible years for "forgiveness of state taxes" (*iyr>yr3*), the severance tax is set to zero.

```
        sevtax(iyr)=
     1  ( mwgc*gprice(1,iyr)*gas_sev(istate)+
     2  mwgc*gas_sev_p(istate) )*(1-royrate)/1000.0
        if (fsttax .and. iyr .le. yr3) sevtax(iyr)=0
```

Step 15: **Depletable G&G and lease acquisition (*dggla()*), adjustment for federal tax credit (*dep_crd()*), and total G&G lease cost depletion (*tgglcd()*) are calculated.**

Note: *dggla()* and *dep_crd()* are calculated based on logical variables *ggctc* (allow G&G depletable tax credit) and *lactc* (allow lease acquisition depletable tax credit) specified in input file TAX_NAT.DAT. Total G&G lease cost depletion (*tgglcd()*) is calculated only if *dggla()* and gas production (*temp*) in the corresponding year are not zero.

```
        dggla(iyr)=gg(iyr)*pggc+la(iyr)*plac
        if(ggctc) then
          dggla(iyr)=dggla(iyr)-gg(iyr)*pggc*ggctcr
          dep_crd(iyr)=dep_crd(iyr)+gg(iyr)*pggc*ggctcr
        endif
        if(lactc) then
          dggla(iyr)=dggla(iyr)-la(iyr)*plac*lactcr
          dep_crd(iyr)=dep_crd(iyr)+la(iyr)*plac*lactcr
        endif
        temp=0.0
        do iyr1=iyr,nyr
```

```
          temp=temp+mwgc/5.642/1000.0
        enddo
        if(dggla(iyr).ne.0 .and. temp .ne.0) then
          do iyr1=iyr,nyr
            tgglcd(iyr1)=tgglcd(iyr1)+
     &      dggla(iyr)*(mwgc/5.642/1000.0)/temp
          enddo
        endif
```

Step 16: **Allowable percent depletion (*apd()*) is calculated.**

Note: *apd()* is calculated based on logical variable *nil* (allow net income limitation) which is specified in input file TAX_NAT.DAT.

```
        nilb(iyr)=netsales(iyr)-sevtax(iyr)-toc(iyr)-
     &  ii(iyr)+intcap(iyr)-eggla(iyr)-depr(iyr)
        if(nil) then
          if(nilb(iyr).gt.0) then
            apd(iyr)=min(nilb(iyr)*nill,netsales(iyr)*pdr)
          else
            apd(iyr)=0.
          endif
        else
          apd(iyr)=netsales(iyr)*pdr
        endif
```

Step 17: **Depletion (*deplet()*) is set to the higher value between total G&G lease cost depletion (*tgglcd()*) and allowable percent depletion (*apd()*).**

```
        deplet(iyr)=max(tgglcd(iyr),apd(iyr))
```

Step 18: **Net income before tax addback (*nibta()*) is calculated.**

Note: “eortc” is the logical flag for the EOR tax credit, “eortca” is the EOR tax credit addback, “eortcr” is the EOR tax credit rate.

```
      if(eortc) eortca(iyr)=eortcr*(ii(iyr)-intcap(iyr)+inj(iyr))
      nibta(iyr)=
   1  netsales(iyr)-sevtax(iyr)-toc(iyr)-ii(iyr)+intcap(iyr)-
   2  eggla(iyr)-depr(iyr)-deplet(iyr)
```

Step 19: **Intangible drilling cost addback (*idca()*) is calculated.**

Note: *idca()* is calculated based on logical variables *tcoii* (allow tax credit on intangible investments), *cidc* (intangible drilling costs to be capitalized), *idctc* (allow intangible drilling cost tax credit), and *cidc* (intangible drilling costs to be capitalized) specified in input file TAX_NAT.DAT.

```
        if(tcoii) then
          if(yr2.ge.iyr) then
            if(cidc) then
              idca(iyr)=(1-piic)*(intang_dwc(iyr)+
     1                   intang_ewc(iyr))*idctcr
            else
              idca(iyr)=(intang_dwc(iyr)+intang_ewc(iyr))*idctcr
            endif
          else
            idca(iyr)=0
          endif
        else
          if(idctc) then
            if(cidc) then
              idca(iyr)=(1-piic)*(intang_dwc(iyr)+
     1                 intang_ewc(iyr))*idctcr
            else
              idca(iyr)=(intang_dwc(iyr)+intang_ewc(iyr))*idctcr
            endif
          else
            idca(iyr)=0
          endif
        endif
```

Step 20: **Other intangible addback (*oia()*) is calculated.**

Note: *oia()* is calculated based on logical variables *tcoii* (allow tax credit on intangible investments), *coi* (other intangibles to be capitalized), and *oitc* (allow other intangible tax credit) specified in input file TAX_NAT.DAT.

```
        if(tcoii) then
          if(yr2.ge.iyr) then
            if(coi) then
              oia(iyr)=(1-piic)*icap(iyr)*oitcr
            else
              oia(iyr)=icap(iyr)*oitcr
            endif
          else
            oia(iyr)=0
          endif
        elseif(oitc) then
          if(coi) then
            oia(iyr)=(1-piic)*icap(iyr)*oitcr
          else
            oia(iyr)=icap(iyr)*oitcr
          endif
        else
          oia(iyr)=0
        endif
```

Step 21: **Intangible environmental addback (*iea()*) is calculated.**

Note: *iea()* is set to zero if environmental intangible tax credit is not allowed (*eitc=.false.*, specified in file TAX_NAT.DAT). *iea()* is calculated based on logical variable *ce* (environmental to be capitalized) specified in input file TAX_NAT.DAT.

```
            if(eitc) then
              if(ce) then
                iea(iyr)=(1-piic)*eicap(iyr)*eitcr
              else
                iea(iyr)=eicap(iyr)*eitcr
              endif
            else
              iea(iyr)=0
            endif
```

Step 22: **Environmental operating cost addback (*eoca()*) is calculated.**

Note: *eoca()* is set to zero if environmental operating cost tax credit is not allowed (*eoctc=.false.*, specified in file TAX_NAT.DAT). Otherwise, it is set equal to environmental operating and maintenance cost (*eoam()*) multiplied by environmental operating cost tax credit rate (*eoctcr*).

```
            if(eoctc) then
              eoca(iyr)=eoam(iyr)*eoctcr
            else
              eoca(iyr)=0
            endif
```

Step 23: **G&G/lease addback (*ggla()*), total intangible addback (*intadd()*), net income before taxes (*nibt()*), state income tax (*sttax()*), and federal taxable income (*fti()*) are calculated.**

Note: *ggla()* is calculated based on logical variables *ggetc* (allow tax credit for expensed G&G) and *laetc* (allow tax credit for expensed lease acquisition costs) specified in file TAX_NAT.DAT). If forgiveness of state taxes is allowed (*fsttax=.true.*, specified in input file TAX_NAT.DAT), and within the eligible years for "forgiveness of state taxes" (*yr3>=iyr*), and *nibt()* greater than zero, the state income tax (*sttax()*) is set to zero.

```
            if(ggetc) then
              ggla(iyr)=ggla(iyr)+ggetcr*gg(iyr)*(1-pggc)
            endif
            if(laetc) then
              ggla(iyr)=ggla(iyr)+laetcr*la(iyr)*(1-plac)
            endif
            intadd(iyr)= idca(iyr)+oia(iyr)+iea(iyr)+eoca(iyr)
            nibt(iyr)  = nibta(iyr)+eortca(iyr)+intadd(iyr)+ggla(iyr)
            if(fsttax .and. yr3.ge.iyr .and. nibt(iyr).gt.0) then
              sttax(iyr)=0
            else
              sttax(iyr)=nibt(iyr)*strate(istate)
            endif
            fti(iyr)=nibt(iyr)-sttax(iyr)
```

Step 24: **Excess intangible drilling cost addback (*eidca()*), net income from oil and gas (*nifoag()*), intangible drilling cost preference for alternative minimum taxable (*idcpamt()*), unadjusted and adjusted alternative minimum taxable incomes (*uamti()* and *amti()*), ACE and ACE adjustment (*ace()* and *aceadj()*), alternative minimum taxes (*amint()*), tentative and selected federal income taxes (*tfit()* and *sfit()*), availible and usable credits for past alternative minimum taxable (*acpamt()* and *ucpamt()*), federal income tax (*fedtax()*), and balance of alternative minimum taxable paid (*bamtp()*) are calculated.**

Note: *ip* is logical variable for independent producer.

```
         eidca(iyr)   = (1-smar)*(ii(iyr)-intcap(iyr))
         nifoag(iyr)  = fti(iyr)+eidca(iyr)
         if(nifoag(iyr).gt.0) dpidcs(iyr)=nifoag(iyr)*ipd
         idcpamt(iyr) = eidca(iyr)-dpidcs(iyr)
         if(ip) then
          uamti(iyr)=max(fti(iyr),(1-ira)*(fti(iyr)+idcpamt(iyr)))
         else
           uamti(iyr)=fti(iyr)+idcpamt(iyr)
         endif
         if(.not. ip) then
           aceadj(iyr)=dpidcs(iyr)
           if(deplet(iyr).gt.tgglcd(iyr))
     &     aceadj(iyr)=deplet(iyr)-tgglcd(iyr)
         endif
         ace(iyr)=uamti(iyr) + aceadj(iyr)
         if(ace(iyr) .gt. uamti(iyr)) then
           amti(iyr)=uamti(iyr)+acer*(ace(iyr)-uamti(iyr))
         else
           amti(iyr)=uamti(iyr)
         endif
         amint(iyr)=amtrate*amti(iyr)
         tfit(iyr)=(nibt(iyr)-sttax(iyr))*fedrate
         if(amt) then
           sfit(iyr)=max(amint(iyr),tfit(iyr))
         else
           sfit(iyr)=tfit(iyr)
         endif
         if(iyr.eq.1) then
           acpamt(iyr)=0
         else
           acpamt(iyr)=bamtp(iyr-1)
         endif
         if(tfit(iyr) .gt. amint(iyr) .and. credamt) then
           ucpamt(iyr)=min(acpamt(iyr),tfit(iyr)-amint(iyr))
         else
           ucpamt(iyr)=0
         endif
         fedtax(iyr)=sfit(iyr)-ucpamt(iyr)
         if(iyr.eq.1) then
           bamtp(iyr)=fedtax(iyr)-tfit(iyr)
         else
           bamtp(iyr)=bamtp(iyr-1)+fedtax(iyr)-tfit(iyr)
         endif
```

Step 25: **Federal tax credits (*fedtaxc()*) is calculated.**

Note: *fedtaxc()* is calculated based on logical variables *eortc* (allow enhanced oil recovery tax credit), *tcoti* (allow tax credit on tangible investments), *ggctc* (allow G&G depletable tax credit), *ggetc* (allow tax credit for expensed G&G), *lactc* (allow lease acquisition depletable tax credit), *laetc* (allow tax credit for expensed lease acquisition costs), *tdtc* (allow tangible development tax credit), *ettc* (allow environmental tangible tax credit), *tcoii* (allow tax credit on intangible investments), *idctc* (allow intangible drilling cost tax credit), *oitc* (allow other intangible tax credit), *eitc* (allow environmental intangible tax credit), and *eoctc* (allow environmental operating cost tax credit). Except for *eortc*, YES/NO responses for all of these logical variables are obtained from input file TAX_NAT.DAT.

```
   if(eortc) fedtaxc(iyr)=fedtaxc(iyr)+
 &  eortcr*(ti(iyr)+ii(iyr)+inj(iyr))
   if(tcoti) then
     if(yr1.ge.iyr) then
       fedtaxc(iyr)=fedtaxc(iyr)+ggctcr*gg(iyr)*pggc
       fedtaxc(iyr)=fedtaxc(iyr)+lactcr*la(iyr)*plac
       fedtaxc(iyr)=fedtaxc(iyr)+tdtcr*(tang_dwc(iyr)+
1                   tang_ewc(iyr))
       fedtaxc(iyr)=fedtaxc(iyr)+ettcr*etcap(iyr)
    endif
   else
     if(ggctc) fedtaxc(iyr)=fedtaxc(iyr)+ggctcr*gg(iyr)*pggc
     if(ggetc) fedtaxc(iyr)=fedtaxc(iyr)+ggetcr*gg(iyr)*(1-pggc)
     if(lactc) fedtaxc(iyr)=fedtaxc(iyr)+lactcr*la(iyr)*plac
     if(laetc) fedtaxc(iyr)=fedtaxc(iyr)+laetcr*la(iyr)*(1-plac)
     if(tdtc)  fedtaxc(iyr)=fedtaxc(iyr)+tdtcr*(tang_dwc(iyr)+
1                           tang_ewc(iyr))
     if(ettc) fedtaxc(iyr)=fedtaxc(iyr)+ettcr*etcap(iyr)
   endif
   if(tcoii) then
   if(yr2.ge.iyr) then
       fedtaxc(iyr)=fedtaxc(iyr)+idctcr*(intang_dwc(iyr)+
1                   intang_ewc(iyr))
       fedtaxc(iyr)=fedtaxc(iyr)+ oitcr*icap(iyr)
     endif
   else
     if(idctc) fedtaxc(iyr)=fedtaxc(iyr)+
&                      idctcr*(intang_dwc(iyr)+intang_ewc(iyr))
     if(oitc) fedtaxc(iyr)=fedtaxc(iyr)+ oitcr*icap(iyr)
  endif
          if(eitc) fedtaxc(iyr)=fedtaxc(iyr)+eitcr*eicap(iyr)
          if(eoctc) fedtaxc(iyr)=fedtaxc(iyr)+eoctcr*eoam(iyr)
```

Step 26: **Net income after taxes (*niat()*), annual after tax cash flow (*aatcf()*), discounted after tax cash flow (*datcf()*), and annual after tax cash flow (*aatcf()*) are calculated.**

Note: "stor_gas_cost" is the cost of base gas.

```
          niat(iyr)=nibt(iyr)-sttax(iyr)-fedtax(iyr)+fedtaxc(iyr)
          aatcf(iyr)=niat(iyr)+depr(iyr)+deplet(iyr)-
       1  dggla(iyr)-intcap(iyr)-ti(iyr)-eortca(iyr)-
       2    intadd(iyr)-ggla(iyr)
          datcf(iyr)=aatcf(iyr)/((1+disc)**(iyr-1))
          if(iyr.eq.1) catcf(iyr)=datcf(iyr)
          if(iyr.gt.1) catcf(iyr)=catcf(iyr-1)+datcf(iyr)
         numer = numer+( stor_gas_cost(iyr)+otc(iyr)+tang_dwc(iyr)+
    @     intang_dwc(iyr)+la(iyr))/( (1.0+disc)**(iyr-1) )
         denom = denom + mwgc/( (1.0+disc)**(iyr-1) )
     enddo !iyr
       if (denom .gt. 0.01) then
         lcst_tmp=numer/(denom/1000.0)
       else
         lcst_tmp=0.0
       end if
       return
       end
```

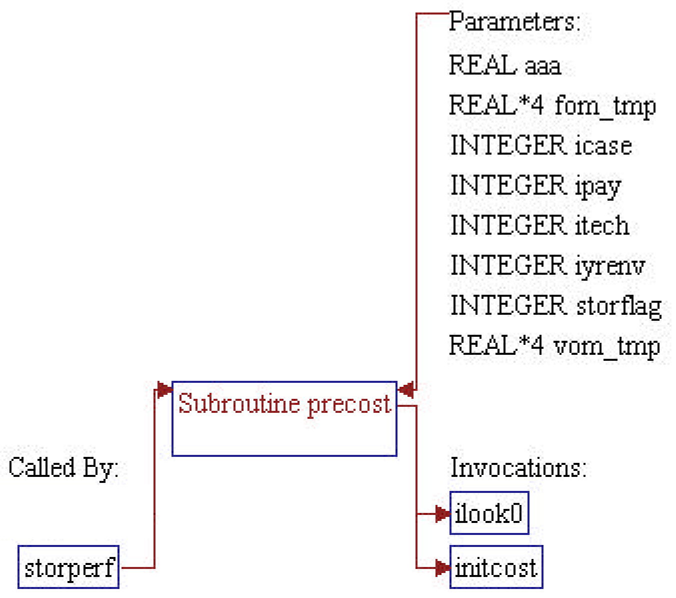
SUB-PROGRAM PRECOST()

MAIN THEME: This routine utilizes the unit cost data to create the cost streams to be fed to the cash flow routine CASHFLOW().

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameter of the subroutine:

• *itech*	Technology flag (should be 1 for current technology)
• *icase*	Development case flag (should be 1 for primary wells)
• *iyrenv*	Number of years for environmental run (years)
• *storflag*	Flag for storage (full production mode should be1)
• *fom_tmp*	Fixed O&M cost
• *vom_tmp*	Variable O&M cost
• *ipay*	Pay grade number
• *aaa*	Total surface O&M costs

```
      subroutine precost(itech,icase,iyrenv,storflag,fom_tmp,
     &       vom_tmp,ipay,aaa)
```

Note: Include files.

```
      include 'dimen.h'
      include 'global.h'
      include 'rd_data.h'
      include 'cost.h'
      include 'field.h'
      include 'costing.h'
      include 'tax_nat.h'
      include 'tax_reg.h'
      include 'unitcost.h'
      include 'welldata.h'
      include 'type_out.h'
      include 'gsamvar.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
      include 'storlp.h'
```

Note: Common block and local variables.

```
      common /stchg/ iwin_yr
      integer iyr,itech,icase
      integer comp_yr
      integer winyr,iyrenv,ipay
      integer storflag
      real*4 fxoam(qyr)
      real*4 voam(qyr)
      real*4 h2ovoam(qyr)
      real*4 faccost(qyr)
      real*4 oam_comp(qyr)
      real*4 fom_tmp,vom_tmp
      real*4 denom
      real*4  new_well , old_well
```

Step 2: **Fixed operating and maintenance (O&M) cost (*fxoam()*), surface gas O&M cost (*voam()*), surface water O&M cost (*h2ovoam()*), facilities cost (*faccost()*), and compressor O&M cost (*oam_comp()*) in each year are initialized to zero.**

```
      denom = 0.0
      fom_tmp = 0.0
      vom_tmp = 0.0
      aaa = 0.0
      do iyr=1,qyr
       fxoam(iyr)    = 0.0
       voam(iyr)     = 0.0
       h2ovoam(iyr)  = 0.0
       faccost(iyr)  = 0.0
       oam_comp(iyr) = 0.0
       stor_gas(iyr)  = 0.0
       stor_gas_cost(iyr) = 0.0
       oam(iyr)       = 0.0
     enddo
```

Step 3: **Sub-program INITCOST is invoked to initialized other costing variables.**

Note: These variables are declared in header file COSTING.H.

```
      call initcost
```

Step 4: **Compression is added in the first year for storage projects**

```
        comp_yr=1
```

Step 5: **The logic for calculating drilling well costs in SRPM is as follows:**
a) New wells are only drilled when required after stimualting all existing wells.
b) The number of new wells to be drilled is calculated using the following formula: number of wells to be drilled = Total Number of wells used in SRPM by paygrade based on wspace and area (nwell) – Reported number of wells from American Gas Association Survey (dbwells).

‘io_wells’ refers to the existing wells in New/Old fields in a particular paygrade.
Hence if (nwell - dbwells) <= 0 then nwells are stimulated and no new drilling occurs, however, if (nwell - dbwells) > 0 then dbwells are stimulated and (nwell - dbwells) wells are drilled and costs are calculated.

Here new wells needed are calculated.

```
if (io_wells(ipay).ge.nwell) then
     old_well = nwell
     new_well = 0.0
 else
     old_well =io_wells(ipay)
     new_well =nwell - io_wells(ipay)
endif
```

Step 6: **Drilling well cost is calculated for new wells only.**

```
  dwc(1) = dwc(1)+((dwc_w)+(prob_dry(itech)*dwc_w*
&          pdry_dev(itech)))*new_well
```

Step 6: **Stimulation cost for OLD+NEW wells is calculated in year 1.**

```
stim(1)=stim(1)+stim_w*(new_well+ old_well)*intang_m(1)
```

Step 7: **Fixing cost for existing wells only is calculated.**

```
 fix(1) = fix_w*io_wells(ipay)
do iyr = 2,nyrset_storage
  stim(iyr) = 0.0
enddo
```

Step 8: **Re-stimulation cost is calculated. For depleted gas (module - 7) and aquifers (module - 8) the wells are stimulated after every five years for salt dome (module - 9) the wells are stimulated every other year to maintain deliverabilty.**

```
interval = 1
if (module.eq.7.or.module.eq.8) interval = 5
if (module.eq.9) interval = 2
do iyr = 1+interval,nyrset_storage,interval
  stim(iyr)=stim(iyr)+stim_w*(new_well+ old_well)*intang_m(iyr)
enddo
```

Step 9: **Compressor installation cost is calculated.**

```
  comp(comp_yr)=comp_w*(new_well+ old_well)*tang_m(comp_yr)
```

Step 10: **Facility cost is calculated.**

```
  faccost(1) = fac_w*(new_well + old_well)
```

Step 11: **Base gas cost is set to zero. The demand and integrating module will calculate this out separately.**

```
  stor_gas_cost(1) = 0.0
```

Step 12: **Fixed O&M cost is calculated.**

```
do iyr=1,nyrset_storage
  fxoam(iyr)=(new_well + old_well)*fxoam_w
enddo
```

Step 13: **Lease bonus cost is calculated using $5/acre assumption.**

```
 la(1)   = 25.0*area(ipay)/1000000.0
do 96 iyr=1,nyrset_storage
```

Step 14: **Annual lease rental is calculated, $5 per acre.**

```
  la_oam(iyr) = 5.0*area(ipay)/1000000.0
```

Step 15: **Surface O&M costs are calculated.**

```
  voam(iyr)=(mwgc)/1000.0*voam_g
```

Step 16: **The cost to run the compressor to inject gas ($0.015/Mcf Injected) is calculated.**

```
            oam_comp(iyr) = gasinje(iyr)*0.015
```

Step 17: Total o&m is calculated.

```
          oam(iyr)= ( la_oam(iyr)+
     &    fxoam(iyr)+voam(iyr)+h2ovoam(iyr)+oam_comp(iyr))*oam_m(iyr)
```

Step 18: Fixed and variable o&m is calculated for use in *.SRO

```
         fom_tmp = fom_tmp+(fxoam(iyr)+stim(iyr)
     @         +la_oam(iyr))*oam_m(iyr)/( (1.0+disc)**(iyr-1) )
         vom_tmp = vom_tmp + (h2ovoam(iyr)+voam(iyr)+oam_comp(iyr))*
     &          oam_m(iyr)/( (1.0+disc)**(iyr-1) )
          aaa = aaa + voam(iyr)
```

Step 19: Total environmental O&M is calculated.

```
            eoam(iyr)=
     &    (h2oprod(iyr)*env_oam_w+mwgc/1000.0*env_oam_g)*oam_m(iyr)
```

Step 20: Other tangible cost are calcuated.

```
           otc(iyr)=faccost(iyr)*fac_tan(itech)*tang_m(iyr)+comp(iyr)
```

Step 21: Other intangible capital costs are calcuated.

```
            icap(iyr)=faccost(iyr)*(1-fac_tan(itech))*intang_m(iyr)
```

Step 22: NPV of gas handled for total nyrset_storage years is calculated.

```
            denom = denom+(2+comp_fs)*mwgc/1000.0
     @   /((1.0+disc)**(iyr-1))
   96   continue
```

Step 23: Average fixed o&m and variable o&m cost values are calculated.

```
           if (denom .gt. 0.01) then
                fom_tmp = fom_tmp /denom
                vom_tmp = vom_tmp /denom
             else
                fom_tmp = 0.01
                vom_tmp = 0.01
           endif
```

Step 24: **Environmental tangible capital cost (etcap), environmental intangible capital cost (eicap), and environmental O&M cost (eoam) are calculated.**

```
      etcap(1)= fac_tan(itech)*env_cap_w*tang_m(1)
      iyrenv = 1
      eicap(1)=(1-fac_tan(itech))*env_cap_w*tang_m(1)
      if (iyrenv.gt.0.and.iyrenv.le.nyr) then
        if (iyrenv.le.1) then
          eicap(1) = eicap(1) + envni*nwell*(1.0 + prob_dry(itech))
          etcap(1) = etcap(1) + envnt*nwell
          do iyr = 1, nyrset_storage
            eoam(iyr) = eoam(iyr) + env_oam_n*nwell
          enddo
        else       ! iyrenv.le.1
          eicap(iyrenv) = eicap(iyrenv) + envei*nwell
          etcap(iyrenv) = etcap(iyrenv) + envet*nwell
          do iyr = iyrenv,nyrset_storage
            eoam(iyr) = eoam(iyr) + env_oam_l*nwell
          enddo
        endif       ! iyrenv.le.1
      endif         ! iyrenv.gt.0.and.iyrenv.le.nyr
      return
      end
```

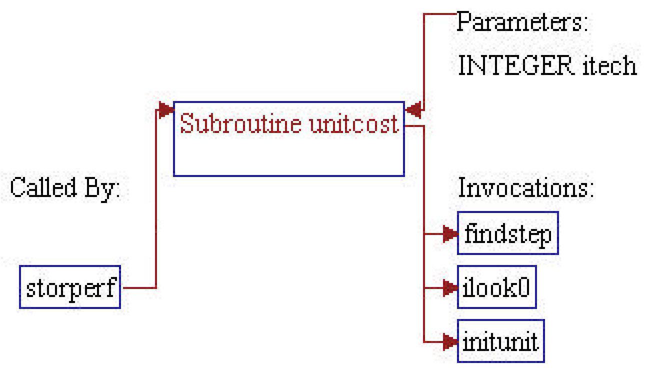
SUB-PROGRAM UNITCOST()

MAIN THEME: This routine calculates per unit costs in $/MCF, $/Well and/or $/BBL.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameter of the subroutine:

- *itech* Technology flag (should be 1 for current technology only)

```
      subroutine unitcost(itech)
```

Note: Include files and local variables.

```
      include 'dimen.h'
      include 'rd_data.h'
      include 'global.h'
      include 'field.h'
      include 'cost.h'
      include 'unitcost.h'
      include 'tax_nat.h'
      include 'gsamvar.h'
      include 'welldata.h'
      include 'type_out.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
      include 'storlp.h'
      integer istep,findstep,iyr,itech
```

Step 2: **Sub-program INITUNIT is invoked to initialize unit cost variables declared in header file UNITCOST.H.**

```
      call initunit
```

Step 3: **Stimulation cost (*stim_w*) for vertical well (*jtyp()=0*) or horizontal well (*jtyp()=1*) is calculated.**

Note: Pressure *pressin* is set to minimum wellhead pressure *premin*. *avdep* is depth to the center of the reservoir, *halfln()* is fracture half length, *netpay* is net pay thickness, *stim_fac()* is development well cost for stimulation length, and *1/1E6* is conversion factor from $ to MM$. Stimulation cost is $20,000 + $2.5/foot + fracturing cost.
Variable *stimfac* is stimulation efficiency. For horizontal wells no fracturing is assumed.

```
    prssin=premin
     if (jtyp(1,1).ge.1) then
       stim_w = (20000 + 2.5*avdep)/1.e6
     else
       stim_w
    1  =(20000+2.5*avdep+(1.5*halfln(2,1)*netpay))/stimfac(itech)/1.e6
     endif
```

Step 4: **Compressor installation cost (*comp_w*) is calculated.**

Note: *peakrate1* is peak production rate and *peakrate2* is injection rate. Compressor cost is assumed to be $1200/BHP.

```
            if (peakrate1.ge.peakrate2) then
             comp_w= (22*2*(peakrate1/1000)*(1000/prssin)**0.50)
             comp_w  = comp_w*1200/1e6/0.95
            else
             comp_w= (22*2*(peakrate2/1000)*(presin/1000)**0.50)
             comp_w  = comp_w*1200/1e6/0.95
            endif
```

Step 5: **Development well unit cost (*dwc_w*), and exploratory well unit cost (*ewc_w*) are calculated. Environmental costs are calculated/assigned.**

Note: Location of data for development well cost in input file COST.DAT (*ireg*) is searched. Sequential searching technique is performed until the region number in the first column of the data (*dwc_reg()*) matches the GSAM supply region given in variable *gsamsr*. Number of data for development well cost calculation (excluding the default data) is *ndwcreg()*. If no match is found (*ireg* is equal to *ndwcreg()*), default data specified in location number *qreg+1* is utilized.

```
           do ireg=1,ndwcreg(itech)
             if(gsamsr.eq.dwc_reg(itech,ireg)) goto 39234
           enddo
           ireg=qreg+1
     39234 continue
```

Note: Location of data for environmental costs in input file COST.DAT (*iregst*) is searched using the same technique as in the development well cost data. The environmental costs data in input file COST.DAT can be entered based on GSAM supply region (number of data is at most 40) or based on State/District (number of data greater than 40). In the following code, number of data (*newcreg()*) is used to identify whether the data is based on GSAM supply region or State. The searching algorithm will use GSAM supply region (*gsamsr*) if *newcreg()<=40*, or State/District code (*state*) if *newcreg()<=40*. The searching procedure is stopped when the ID number in the first column of the data (*ewc_reg()*) matches the value in *gsamsr* or *state.* If no match is found (*iregst* is equal to *newcreg()*), default data specified in location number *newcreg()+1* is utilized.

```
      if(newcreg(itech).gt.40) then
        do iregst=1,newcreg(itech)
          if(state.eq.ewc_reg(itech,iregst)) goto 39235
        enddo
        iregst=newcreg(itech)+1
        goto 39235
      endif
      do iregst=1,newcreg(itech)
        if(gsamsr.eq.ewc_reg(itech,iregst)) goto 39235
      enddo
      iregst=newcreg(itech)+1
```

Note: Development well unit cost (*dwc_w*) is calculated using a 4-order polynomial equation that fits the historical cost versus depth data from the 1997 JAS Survey. Four coefficients given in the development well cost data designated by pointer *ireg* (*dwck()*, *dwcx()*, *dwcxx()*, and *dwcxxx()*) are utilized. The calculated development well unit cost is then divided by drilling cost factor (*dcstf()*) which is the last entry in the data in line *ireg*. *1/1000* is a conversion factor from M$ to MM$.

```
39235 dwc_w=dwck(itech,ireg)+dwcx(itech,ireg)*avdep+
     &dwcxx(itech,ireg)*avdep**2+dwcxxx(itech,ireg)*avdep**3
      dwc_w=dwc_w*dcstf(itech,ireg)/1000.0
```

Note: Cost for blowing, completing, etc. is calculated for the storage well.

```
      fix_w = dwc_w*0.30
```

Note: Environmental well cost (*envw*) is the total of new well environmental tangible and intangible capital costs (*env_nt()* and *env_ni()*). These tangible and intangible costs are then stored into variables *envnt* and *envni*, respectively. Existing well environmental tangible and intangible capital costs (*env_nt()* and *env_ni()*) are stored into variables *envnt* and *envni*, respectively. *1/1000* in all the environmental costs is a conversion factor from M$ to MM$.

```
      envw=env_nt(itech,iregst)+env_ni(itech,iregst)
      envw=envw/1000.
      envni=env_ni(itech,iregst)/1000.
      envnt=env_nt(itech,iregst)/1000.
      envei=env_ei(itech,iregst)/1000.
      envet=env_et(itech,iregst)/1000.
```

Note: Exploratory well unit cost (*ewc_w*) is equal to sum of development well cost (*dwc_w*) and stimulation cost (*stim_w*) multiplied by exploratory well cost factor (*ewc_fac()*) obtained from input file COST.DAT.

```
      ewc_w=ewc_fac(itech)*(dwc_w+stim_w)
```

Step 6: **Facilities well unit cost (*fac_w*) is calculated.**

Note: The facilities well cost (*fac_w*) is calculated. *fac_k()* is the facility cost constant factor ($/well), *fac_s()* is facility cost slope factor ($/well/MCFD), *peakrate* is flow rate, and *1/1E6* is conversion factor from $ to MM$.

```
      istep=findstep(peakrate,fac_max(1,itech),fac_n(itech))
      if(istep.eq.1) then
        fac_w = (fac_k(istep,itech) +
     &  fac_s(istep,itech)*peakrate)/1e6
      else
        fac_w = (fac_k(istep,itech) +
     &  fac_s(istep,itech)*(peakrate-fac_max(istep-1,itech)))/1e6
      endif
```

Step 7: **Fixed operating and maintenance well cost (*fxoam_w*) is calculated.**

Note: Similar to the facilities well cost, several number of regions can be entered for fixed O&M cost data, and several number of steps for different depths can be implemented for fixed O&M cost calculation. The following code searches for the region number (*ireg*) and the location of data for the associated step (*istep*) using similar technique in the previous step except that the direction in searching algorithm for *istep* starts from the depth entry before the last step to the beginning of the step. *nreg_fx()* is number of regions, *fxoam_reg()* is region number data, *fxoam_max()* is depth data, and *fxoam_n()* is number of steps.

```
      do ireg=1,nreg_fx(itech)
       if(gsamid(1:2).eq.fxoam_reg(itech,ireg)) goto 39236
      enddo
      ireg=qreg+1
39236 continue
      if(avdep.ge.fxoam_max(fxoam_n(itech,ireg),itech,ireg)) then
        istep=fxoam_n(itech,ireg)
        goto 121
      else
        do istep=fxoam_n(itech,ireg)-1,1,-1
          if(avdep.ge.fxoam_max(istep,itech,ireg)) goto 121
        enddo
        istep=1
      endif
  121 continue
```

Note: The fixed O&M well cost (*fxoam_w*) is calculated. *fxoam_k()* is the fixed O&M cost constant factor ($/well), *fxoam_s()* is fixed O&M cost slope factor ($/well-ft), *avdep* is reservoir depth, and *1/1E6* is conversion factor from $ to MM$. Two equations are utilized to avoid error due to accessing out of bound array *fxoam_max(istep-1,...,...)* in the case of *istep=1* (reservoir depth is less than the first entry of depth array).

```
       if(istep.eq.1) then
         fxoam_w = (fxoam_k(istep,itech,ireg) +
     &   fxoam_s(istep,itech,ireg)*avdep)/1e6
       else
         fxoam_w = (fxoam_k(istep,itech,ireg) +
     &   fxoam_s(istep,itech,ireg)*
     &   (avdep-fxoam_max(istep-1,itech,ireg)))/1e6
       endif
```

Step 8: **Surface operating and maintenance water cost (*h2ooam_w*) is set equal to the value specified in input file COST.DAT (*oam_h2o()*).**

```
       h2ooam_w=oam_h2o(itech)
```

Step 9: **Variable operating and maintenance gas cost (*voam_g*) is set equal to the sum of operating and maintenance gas cost (*oam_gas()*) and operating and maintenance cost per 1000 feet of well depth (*oam_inc()*avdep/1000*).**

Note: *oam_inc()* is incremental operating and maintenance cost per 1000 feet, *avdep* is reservoir depth, and *1/1000* is used to calculate the incremental factor.

```
       voam_g=oam_gas(itech)+oam_inc(itech)*avdep/1000
```

Step 10: **Lease bonus fraction (*lbc_frac*) which is a fraction of total gas revenues is set equal to lease bonus cost factor specified in input file COST.DAT (*lbc_fac()*).**

```
       lbc_frac=lbc_fac(itech)
```

Step 11: **Environmental capital costs for existing and new wells (*env_cap_w* and *env_cap_n*) are set equal to the facilities well unit cost (*fac_w*) multiplied with environmental capital cost multiplier (*eccm()*) specified in input file COST.DAT.**

```
        env_cap_w=eccm(itech)*fac_w
        env_cap_n=eccm(itech)*fac_w
```

Step 12: **Environmental operating and maintenance costs for gas and water (*env_oam_g* and *env_oam_w*) are set equal to user specified data in input file COST.DAT (*env_g* and *env_w*).**

```
        env_oam_g=env_g(itech,iregst)
        env_oam_w=env_w(itech,iregst)
```

Step 13: **Environmental operating and maintenance costs for existing and new wells (*env_oam_l* and *env_oam_n*) are set equal to the user specified data in input file COST.DAT (*env_ee* and *env_ne*).**

```
        env_oam_l=env_ee(itech,iregst)/1e3
        env_oam_n=env_ne(itech,iregst)/1e3
```

Step 14: **Tangible cost multiplier (*tang_m()*), intangible cost multiplier (*intang_m()*), and operating and maintenance multiplier (*oam_m()*) for gas in each year are calculated.**

Note: *gprice()* is gas price ($/MCF).

```
      do iyr=1,qyr
        tang_m(iyr)   = 1+0.3*(gprice(2,iyr)-2.)/2.
        intang_m(iyr) = 1+0.4*(gprice(2,iyr)-2.)/2.
        oam_m(iyr)    = 1+0.2*(gprice(2,iyr)-2.)/2.
      enddo
      return
      end
```

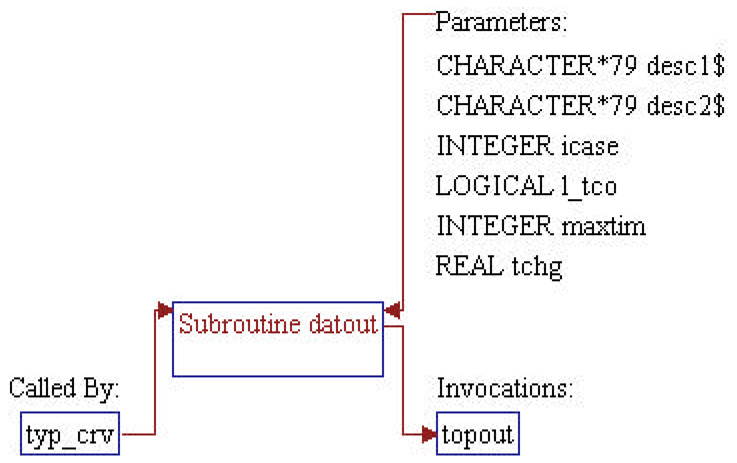
SUB-PROGRAM DATOUT()

MAIN THEME: This routine prints out results to type curve output files (.TCO files) as requested in input file REGIONS.DAT.

READS: None

CREATES: *.TCO

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *desc1$*	First line of description
• *desc2$*	Second line of description
• *maxtim*	Number of time steps
• *icase*	Case number (should be 1 for primary well only)
• *tchg*	Not currently used
• *l_tco*	Logical flag for creating output file *.TCO (unit number 505)

```
      subroutine datout (desc1$,desc2$,maxtim,icase,tchg,l_tco)
```

Note: Include files and local variables.

```
      include 'dimen.h'
      include 'welldata.h'
      include 'type_out.h'
      include 'type1.h'
      include 'type2.h'
      include 'type3.h'
      include 'type4.h'
      include 'type5.h'
      include 'type6.h'
      include 'type7.h'
      include 'type8.h'
      include 'type9.h'
      include 'type10.h'
      include 'rd_data.h'
      include 'gsamvar.h'
      real*4 arr(3),mwgc,mbgc
      character*79 desc1$, desc2$
      logical l_tco
```

Step 2: **The file pointer is rewound and the .TCO file (the .TCO file is the file to which the type curve results are printed) of the previous iteration is overwritten.**

```
      if (l_tco) then
        rewind(unit=505)
      else
        return
      endif
```

Step 3: **“type_work” and “type_base” are calculated.**
"mwgc" is maximum total working gas capacity (MMCF).
"mbgc" is minimum total base gas capacity (MMCF).
"type_work" is working gas capacity in a pay grade (MMCF).
"type_base" is base gas capacity in a pay grade (MMCF).

"type_well" is number of wells in a pay grade.
“cumgas” is cumulative production, (MCF/well).
"qrate" is total production rate in a pay grade (MMCF/D).
"cumpay" is cumulative production in a pay grade (MMCF)
"togip" is total OGIP in the play (MMCF)

```
          mwgc = 0.0
          mbgc = 0.0
          do i = 1,3
            j = 1
            type_work(j,i) = cumgas(i,j,maxtim)*type_well(j,i)/1000.0
            type_base(j,i) = ogip1(i)*type_well(j,i)/1000.0-
     &        type_work(j,i)
            mwgc = mwgc+type_work(j,i)
            mbgc = mbgc+type_base(j,i)
          enddo
```

Step 4: **Peak rate "peakrate" (MMCF/D) is calculated using a quadratic fit extrapolation of flow rates at time steps 1,2, and 3, to get the flow rate at time 0 ("arr"). “gg” is gas flow rate (MCFD/well).**

```
            do i = 1,3
              j = 1
              x1 = time(1)
              x2 = time(2)
              x3 = time(3)
              y1 = qg(i,j,1)
              y2 = qg(i,j,2)
              y3 = qg(i,j,3)
              aa = (y1-2.0*y2+y3)/(2.0*(x1*x1-x2*x2)-(x1*x1-X3*x3))
              bb = (y1-y2-(x1*x1-x2*x2)*aa)/(x1-x2)
              cc = y1-x1*x1*aa-x1*bb
              arr(i) = cc*type_well(j,i)/1000.0
            enddo
            peakrate = max(arr(1),arr(2),arr(3))
```

Step 5: **The following variables are written to the .TCO file:**
Storage ID, Field Name, Reservoir Name, Storage Type, OGIP (MMCF), Working Gas (MMCF), Base Gas (MMCF), Maximum Deliverability (MMCF/D).

Note: “statin” is the flag for storage type: 0 = existing storage, 1 = potential storage.

```
        write(505,'(a35,a40)') 'GSAM ID',resid
        write(505,'(a35,a40)') 'Field Name',fld
        write(505,'(a35,a40)') 'Reservoir Name',resv
        if (statin.eq.0) then
          write(505,'(a35,a40)') 'Storage Type','Existing Storage'
          write(505,*)
          write(505,*)
          write(505,'(a35,2a10)') 'PARAMETER','AGA DATA','ADJ/CALC'
          write(505,*)
          write(505,'(a35,2f10.0)') 'OGIP (MMCF)',capacity,togip
          write(505,'(a35,2f10.0)') 'Working Gas (MMCF)',totwork,mwgc
          write(505,'(a35,2f10.0)') 'Base Gas (MMCF)',totbase,mbgc
          write(505,'(a35,2f10.0)') 'Maximum Deliverability (MMCF/D)',
     &      maxdeliv,peakrate
        else
          write(505,'(a35,a40)') 'Storage Type','Potential Storage'
          write(505,*)
          write(505,*)
          write(505,'(a35,2a10)') 'PARAMETER','AGA DATA','ADJ/CALC'
          write(505,*)
          write(505,'(a35,a10,f10.0)') 'OGIP (MMCF)','N/A',togip
          write(505,'(a35,a10,f10.0)') 'Working Gas (MMCF)','N/A',mwgc
          write(505,'(a35,a10,f10.0)') 'Base Gas (MMCF)','N/A',mbgc
          write(505,'(a35,a10,f10.0)')
     &      'Maximum Deliverability (MMCF/D)','N/A',peakrate
        endif
```

Step 6: **The following variables (for each paygrade) are written to the .TCO file:**
Pay Grade #, Number of Wells, Area (acres), Well Spacing (acres), Thickness (ft), Porosity (%), Permeability (md), Time (time step in years and days), Rate (gas production rate in MMCF/D), Cumulative Production (MMCF), Absolute open flow (MMCFD), Psf (surface pressure in psia), Pwf (well head pressure in psia).

```
      do i = 1,3
        j = 1
        if (area(i).gt.1.0) then
          write(505,*)
          write(505,*)
          write(505,*)
          write(505,'(a30,i15)') 'Pay Grade #',i
          write(505,'(a30,i15)') 'Number of Wells',
     &      nint(type_well(j,i))
          write(505,'(a30,f15.0)') 'Area (acres)',area(i)
          write(505,'(a30,f15.0)') 'Well Spacing (acres)',wspace(i)
          write(505,'(a30,f15.0)') 'Thickness (ft)',thick(i)
          write(505,'(a30,f15.0)') 'Porosity (%)',poros(i)*100.0
          write(505,'(a30,f15.0)') 'Permeability (md)',perm(i)
          write(505,*)
          write(505,'(7a10)') 'Time','Time','Rate','Cum.Prod',
     &      'AOF','Psf','Pwf'
          write(505,'(7a10)') '(years)','(days)','(MMCF/D)','(MMCF)',
     &      '(MMCF/D)','(psia)','(psia)'
          write(505,*)
               do is = 1,maxtim
                 idays = nint(time(is)*365.0)
                qrat  = qg(i,j,is)*type_well(j,i)/1000.0
                cumg = cumgas(i,j,is)*type_well(j,i)/1000.0
                aof =  caof(i,1,is)*type_well(j,i)/1000.0
                 write(505,'(f10.6,i10,5f10.1)') time(is),idays,qrat,
     &             cumg,aof,prbh(i,j,is),prwh(i,j,is)
               enddo
              endif
            enddo
            return
            end
```

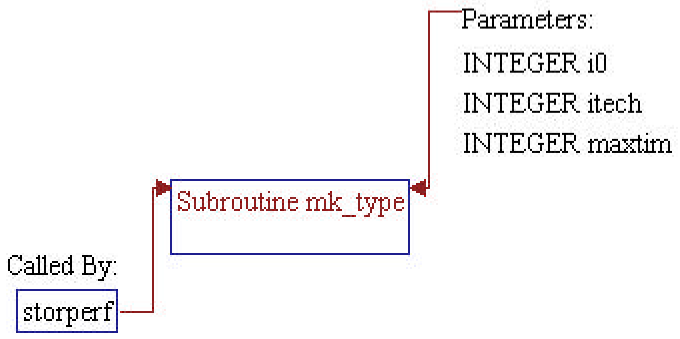
SUB-PROGRAM MK_TYPE()

MAIN THEME: This routine stores type curve input parameters to output file *.TCI.

READS: None

CREATES: *.TCI

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *i0*	Unit number for output file .TCI (unit 506)
• *itech*	Technology flag (should be 1 for current technology only)
• *maxtim*	Number of time steps

```
              subroutine mk_type(i0,itech,maxtim)
```

Note: Include files, common block, and local variables.

```
                include 'dimen.h'
                include 'global.h'
                include 'cost.h'
                include 'tech.h'
                include 'type1.h'
                include 'type2.h'
                include 'type3.h'
                include 'type4.h'
                include 'type5.h'
                include 'type6.h'
                include 'type7.h'
                include 'type8.h'
                include 'type9.h'
                include 'type10.h'
                include 'gsamvar.h'
                integer i,i0,itech,j
                character*80 lines(qline)
                common/ddd/lines
```

Step 2: **Lines 3 and 4 to be printed to file .TCI are set.**

Note: Entries of array variable *lines()* are previously set in sub-program R_TEMP(). This sub-program reads a user specified template file TEMPLATE.DAT and stores the data in variable array *lines()*. The following code changes entries of *lines(3)* with information such as 11-digit GSAM ID and name of technology (*technm()*). *lines(4)* is set to blank.

```
           lines(3)='GSAM Code: '//gsamid//' Technology: '//technm(itech)
           lines(4)=' '
```

Step 3: **Lines 1 through 9 are printed.**

```
                do i=1,9
                  write(i0,555) lines(i)
```

```
      enddo
```

Step 4: **Information related to impurities concentrations, gas gravity, temperature, tubing diameter, and flag for speedup are printed.**

Note: Variable names (or value) and descriptions of these parameters are as follows:

• *gasgrv1*	Gas specific gravity
• *tem*	Temperature (degree F)
• *cnch2s*	Concentration of hydrogen sulfide (fraction)
• *cncco2*	Concentration of carbon dioxide (fraction)
• *cncn2*	Concentration of nitrogen (fraction)
• *cnch2s*	Concentration of hydrogen sulfide (fraction)
• *diam*	Tubing inside diameter (inches)
• *1*	Value for speedup flag. Value of 1 is defaulted in which indicates a speedup run.

```
      write(i0,250)gasgrv1,tem,cnch2s,cncco2,cncn2,diam,1
  250 format(t4,f6.4,t15,f5.0,t24,
     &  f7.2,t34,f7.2,t44,f7.2,t55,f6.3,t68,i1)
```

Step 5: **Lines 11 through 17 are printed.**

```
      do i=11,17
        write(i0,555) lines(i)
      enddo
```

Step 6: **Basic reservoir parameters/properties are printed.**

Note: Variable names and descriptions of these parameters are as follows:

• *i*	Pay grade number
• *pinit()*	Initial reservoir pressure (psia)
• *perm()*	Horizontal permeability (md)
• *permv()*	Vertical permeability (md)
• *poros()*	Total porosity (fraction)
• *swi()*	Initial water saturation (fraction)
• *thick()*	Net pay thickness (feet)
• *salin()*	Water salinity (ppm)

```
      do i=1,3
        write(i0,600) i,pinit(i),perm(i),permv(i),
   &  poros(i),swi(i),thick(i),salin(i)
      enddo
  600 format(t4,i1,t10,f6.0,t17,f7.2,t25,
   & f7.2,t33,f7.2,t44,f5.2,t55,f6.2,t65,f7.0)
```

Step 7: **Lines 21 through 27 are printed.**

```
      do i=21,27
       write(i0,555) lines(i)
      enddo
```

Step 8: **Fractured reservoir properties are printed.**

Note: Variable names and descriptions of these properties are as follows:

• *i*	Pay grade number
• *permm()*	Matrix permeability (md)
• *porma()*	Matrix porosity (fraction)
• *frcspc()*	(not currently used)

```
      do i=1,3
       write(i0,800) i,permma(i),porma(i),frcspc(i)
      enddo
  800 format(t4,i1,t10,f7.2,t21,f6.4,t32,f5.2)
```

Step 9: **Lines 31 through 38 are printed.**

```
      do i=31,38
       write(i0,555) lines(i)
      enddo
```

Step 10: **Field development information/parameters are printed.**

Note: Variable names and descriptions of these parameters are as follows:

• *i*	Pay grade number
• *depth1()*	Depth (feet)
• *area()*	Well drainage area (acres)
• *wspace()*	Well spacing (acres)
• *imod()*	Reservoir Module flags.
• *rw()*	Wellbore radius.

```
      do i=1,3
       write(i0,900)
     &  i,depth1(i),area(i),wspace(i),(imod(i,j),j=1,3),
     &      (rw(i,j),j=1,3)
      enddo
900    format(t4,i1,t8,f6.0,t15,f8.0,t23,f5.0,
     &    t34,i1,t41,i1,t48,i1,t55,f4.2,t61,f4.2,t68,f4.2)
```

Step 11: **Lines 42 through 49 are printed.**

```
             do i=42,49
               write(i0,555) lines(i)
             enddo
```

Step 12: **Fractured and horizontal well data are printed.**

Note: Variable names (or value) and descriptions of these parameters are as follows:

• *i*	Pay grade number
• *0,0,0*	Well types: 0=vertical (default), 1=horizontal
• *halfln()*	Fracture half length or horizontal well length (feet)
• *cond()*	Fracture conductivity (md-ft)

```
         do i=1,3
           write(i0,950) i,0,0,0,(halfln(i,j),j=1,3),(cond(i,j),j=1,3)
         enddo
  950    format(t4,i1,t10,i1,t17,i1,t24,i1,t31,f5.0,t38,f5.0,t45,f5.0,
      &    t54,f6.0,t61,f6.0,t68,f6.0)
```

Step 13: **Lines 53 through 61 are printed.**

```
             do i=53,61
               write(i0,555) lines(i)
             enddo
```

Step 14: **Parameters for water drive and unconventional reservoir data are printed (not currently used).**

```
          do i=1,3
            write(i0,690) i,kaqtyp(i),sgtrap(i),qwmax(i),
       &      kuncon(i),iloc(i),gascon1(i),pl(i),tdes(i),rhoma(i)
          enddo
   690    format(t4,i1,t07,i2,5x,f4.2,4x,f5.1,6x,i1,10x,i1,4x,f4.0,6x,
       &    f5.0,5x,f4.0,5x,f4.2)
```

Step 15: **Lines 65 through 72 are printed.**

```
             do i=65,72
               write(i0,555) lines(i)
```

```
       enddo
```

Step 16: **Well control parameters are printed.**

Note: Variable names (or value) and descriptions of these parameters are as follows:

• *premin*	User specified minimum wellhead pressure (psia)
• *ratmax*	Maximum gas rate. Units and definition depend on magnitude of *ratmax*: >1.0 (MCFD), <=1.0 (fraction of absolute open flow)
• *timchg*	(not currently used)
• *1*	First pay grade for the following skin factors
• *skin(1,1,1)*	Skin factor for primary well in pay grade 1
• *skin(1,2,1)*	(not currently used)
• *skin(1,3,1)*	(not currently used)
• *skin(1,1,2)*	(not currently used)
• *2*	Second pay grade for the following skin factors
• *skin(2,1,1)*	Skin factor for primary well in pay grade 2
• *skin(2,2,1)*	(not currently used)
• *skin(2,3,1)*	(not currently used)
• *skin(2,1,2)*	(not currently used)
• *3*	Third pay grade for the following skin factors
• *skin(3,1,1)*	Skin factor for primary well in pay grade 3
• *skin(3,2,1)*	(not currently used)
• *skin(3,3,1)*	(not currently used)
• *skin(3,1,2)*	(not currently used)

```
       write(i0,960) premin,ratmax,timchg,1,
    &    skin(1,1,1),skin(1,2,1),skin(1,3,1),skin(1,1,2)
960    format(t3,f6.0,t12,f9.2,t25,f4.1,t36,i1,t42,
    &    f5.1,t48,f5.1,t55,f5.1,t65,f6.1)
       write(i0,970) 2,
    &    skin(2,1,1),skin(2,2,1),skin(2,3,1),skin(2,1,2)
       write(i0,970) 3,
    &    skin(3,1,1),skin(3,2,1),skin(3,3,1),skin(3,1,2)
970    format(t36,i1,t42,f5.1,t48,f5.1,t55,f5.1,t65,f6.1)
555    format(a)
       return
       end
```

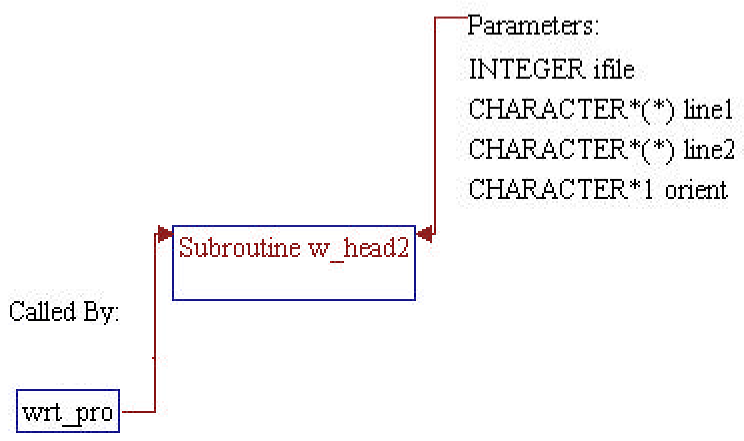
SUB-PROGRAM W_HEAD2()

MAIN THEME: This routine prints out two header lines of a table to a specified output file.

READS: None

CREATES: Variable output file unit number

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutines:

• *ifile*	Output file unit number of the table
• *line1*	The first header line
• *line2*	The second header line
• *orient*	Orientation flag (‘P’=portrait, ‘L’=landscape)

```
      subroutine w_head2(ifile,line1,line2,orient)
```

Note: Local variables.

```
      character*(*) line1,line2
      character*1 orient
      integer ifile
```

Step 2: **Header lines are printed.**

```
      if(orient.eq.'L') then
       write(ifile,*) '_E_&l1o5.45C_&k2S'
      else
       write(ifile,*) '_E_&l5.45C_(0U_(sp16.66h7vsb8T'
      endif
      write(ifile,*) '_&d@_(119X'
      if(line1.ne.' ') write(ifile,1200) line1
      if(line2.ne.' ') write(ifile,1200) line2
      write(ifile,1230)
 1200 format(t30,a)
 1220 format(t30,a,'_(3@_&k2S',/)
 1230 format('_(3@_&k2S',/)
      return
      end
```

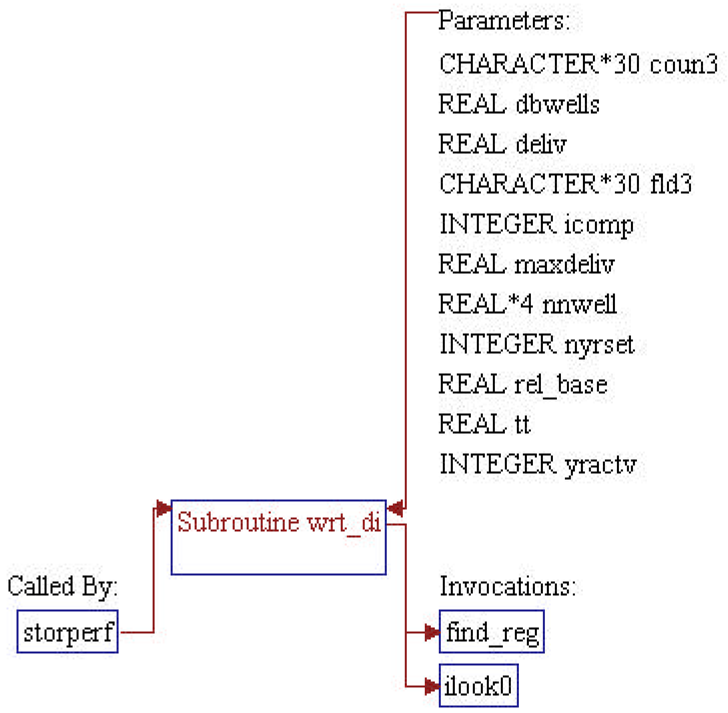
SUB-PROGRAM WRT_DI()

MAIN THEME: This routine writes out reservoir performance data to output file *.SRO, to be used in Demand and Integrating Model.

READS: None

CREATES: *.SRO

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutines:

• *yractv*	The year the reservoir was discovered
• *tt*	NPV of total working gas that will be handled for next *nyrset_storage* years for potential storage reservoir (calculated in CASHFLOW() routine)
• *icomp*	Company ID number

```
            subroutine wrt_di(yractv,tt,icomp)
```

Note: Include files and local variables.

```
            include 'gsamvar.h'
            include 'storlp.h'
            integer      resvfyr,yractv
            character*2  name
            character*15 resvid
            character*20 nname
```

Step 2: **Demand region for the storage reservoir is obtained.**

```
            name  =gsamid(1:2)
            call find_reg(name,nname)
              resvfyr = yractv
            resvid(1:11) = gsamid
            resvid(12:15)= '    '
```

Step 3: **The following calculations are made and variables printed for existing storage.**

```
      IF (mwgc.gt.0.0) THEN
         if (resvid(4:4).eq.'0') then !Existing Storage
           icounterc = 0
           call ilook0(icomp,icompany,n_tot_lev,icounterc)
            if (icounterc.eq.0) then
             icomp = 999
             call ilook0(icomp,icompany,n_tot_lev,icounterc)
            endif
               lcst_op3 = lev_ex(icounterc)
               fom_op3  = fix_ex(icounterc)
               vom_op3 = var_ex(icounterc)
               lcst_op2 = lcst_op3*1.1
               fom_op2  = fom_op3*1.1
               vom_op2 = vom_op3*1.1
               lcst_op1 = lcst_op2*1.1
               fom_op1  = fom_op2*1.1
               vom_op1 = vom_op2*1.1
                     write(503,800) gsamid,resvfyr,mwgc,0.0,20.0,fus,
     &               lcst_op1,fom_op1,vom_op1,mecp_op1_1,mecp_op1_2,
            &               mecp_op1_3,micp,lcst_op2,fom_op2,vom_op2,
     &               mecp_op2_1,mecp_op2_2,micp,lcst_op3,
   &                fom_op3,vom_op3,mecp_op3_1,micp,nname
```

Step 4: **The following calculations are made and variables printed for potential storage.**

```
         else  ! New Storage
              lcst_op3 = lcst1
              fom_op3  = fom
              vom_op3 = vom
              if ((mecp_op1_1-mecp_op1_2).le.0.001.and.
   @              (mecp_op1_1-mecp_op1_3).le.0.001) then
               lcst_op2 = lcst1
               fom_op2  = fom
               vom_op2 = vom
               lcst_op1 = lcst1
               fom_op1  = fom
               vom_op1 = vom
              else
               lcst_op2 = lcst_op3*1.1
               fom_op2  = fom_op3*1.1
               vom_op2 = vom_op3*1.1
               lcst_op1 = lcst_op2*1.1
               fom_op1  = fom_op2*1.1
               vom_op1 = vom_op2*1.1
              endif
      write(503,800) gsamid,resvfyr,mwgc,mbgc/tt,20.0,fus,
   &                lcst_op1,fom_op1,vom_op1,mecp_op1_1,mecp_op1_2,
   &                mecp_op1_3,micp,lcst_op2,fom_op2,vom_op2,
   &                mecp_op2_1,mecp_op2_2,micp,lcst_op3,
   &                fom_op3,vom_op3,mecp_op3_1,micp,nname
        endif
      ENDIF
800 format(a11,2x,i4,1x,f9.1,1x,f6.3,2x,f3.0,1x,f5.2,t47,f6.2,1x,
   1  f6.2,1x,f6.2,1x,f6.2,1x,f6.2,1x,f6.2,1x,f6.2,1x,6(f6.2,1x),
   2  5(f6.2,1x),a20)
820  format(a20,2x,i4,2x,a20,2x,f8.2,2x,f10.2,2x,f15.3)
    return
    end
```

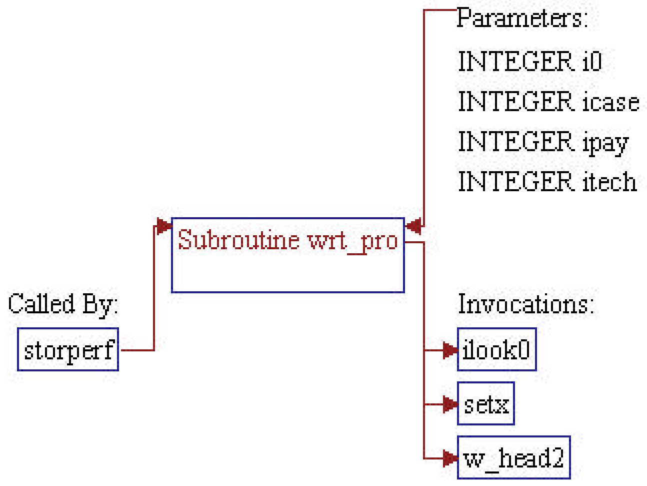
SUB-PROGRAM WRT_PRO()

MAIN THEME: This routine writes out cash flow pro-forma to output file *.PRO.

READS: None

CREATES: *.PRO

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutines:

• *i0*	Unit number for output file .PRO (unit 507)
• *itech*	Technology flag (should be 1 for current technology only)
• *icase*	Case number (should be 1 for current technology only)
• *ipay*	Pay grade number

```
      subroutine wrt_pro(i0,itech,icase,ipay)
```

Note: Include files and local variables.

```
      include 'dimen.h'
      include 'global.h'
      include 'field.h'
      include 'cashflow.h'
      include 'costing.h'
      include 'tax_nat.h'
      include 'tax_reg.h'
      include 'cost.h'
      include 'tech.h'
      include 'gsamvar.h'
      include 'storlp.h'
     integer iyr1,iyr2,i0,numcol,npage,ipage,iyr,itech,icase,ipay
     character*80 line80
     character*2 ch2
     integer nyr1
     real*4 toc_mcf(qyr)
```

Step 2: **Sub-program ILOOK0() is invoked to search for location of region identifier in array *tax_st()* which corresponds to state code *state.***

```
      call ilook0(state,tax_st,ntax_st,istate)
```

Step 3: **Value from pay grade code *ipay* is assigned to 2-digit character variable *ch2*.**

```
      write(ch2,'(i2)') ipay
```

Step 4: **Total operating cost per MCF of gas produced (*toc_mcf()*) is calculated.**

Note: First, sub-program SETX() is invoked to zero out array variable *toc_mcf()*, then the cost is calculated by dividing total operating cost (*toc()*) with total gas production (*gasprod()*).

```
      call setx(toc_mcf,qyr,0)
      do iyr=1,nyr
       if(gasprod(iyr).gt.0) toc_mcf(iyr)=toc(iyr)/gasprod(iyr)
      enddo
```

Step 5: **String variable *line80* is set.**

Note: *line80* is printed as a header line in output file .PRO. Information written to this variable includes 11-digit GSAM ID, name of technology, and pay grade number.

```
      line80=
     & 'GSAM ID: '//gsamid//' Tech.: '//technm(itech)//
     &  ' Case: '//casename(icase)//' P.G.: '//ch2
```

Step 6: **Number of pages to be printed (*npage*) is calculated.**

```
      numcol=7
      nyr1=nyr
      npage=(nyr1)/numcol
      if(mod(nyr1,numcol).gt.0) npage=npage+1
```

Step 7: **Loop for pages is initialized.**

```
      do ipage=1,npage
```

Step 8: **Header lines are printed.**

Note: Sub-program W_HEAD2() is invoked to print the first two header lines. String variable *line80* is passed to W_HEAD2() and printed as the second line. Character *'P'* passed to W_HEAD2() is an indicator to print these header lines with orientation portrait. A word "*Continued*" is added to the first header line if this is not the first page. The beginning and end year numbers (*iyr1* and *iyr2*) for the current page is calculated. The year numbers are then printed to the current page in tabular form.

```
       iyr1=1+numcol*(ipage-1)
       iyr2=min(nyr1,numcol+numcol*(ipage-1))
       if(ipage.eq.1) then
         call w_head2(i0,'Detailed Financial Report',line80,'P')
```

```
           else
             call w_head2(i0,'Detailed Financial Report - Continued',
     &            line80,'P')
           endif
           write(i0,1011) 'Year',(iyr,iyr=iyr1,iyr2)
           write(i0,2000) ('========',iyr=iyr1,iyr2)
```

Step 9: **Cash flow pro-forma is printed.**

Note: Each page of the output file .PRO is a seven column table where the first column is the component's name of the cash flow followed by six values of that component based on years of the current page. The first page will show values of year 1 to year 6.

```
        write(i0,1013) 'Oil Production (MMBO)',
     1    (oilprod(iyr),iyr=iyr1,iyr2)
        write(i0,1013) 'Gas Production (BCF)',
     1    (gasprod(iyr),iyr=iyr1,iyr2)
        write(i0,101) 'Gross Revenues (MM$)',
     1   (oilprod(iyr)*oprice(iyr)+
     2    gasprod(iyr)*gprice(1,iyr),iyr=iyr1,iyr2)
        write(i0,102) 'Gravity/Trans. Cost Adj.',
     1   (gravpen(iyr)+transcst(iyr),iyr=iyr1,iyr2)
        write(i0,101) 'Adjusted Revenues',(adjgross(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'Royalties',
     1     (adjgross(iyr)*royrate,iyr=iyr1,iyr2)
        write(i0,101) 'Net Sales',(netsales(iyr),iyr=iyr1,iyr2)
        write(i0,101) 'Total Operating Cost',(toc(iyr),iyr=iyr1,iyr2)
        write(i0,101) 'Operating Cost/Mcf',
     1    (toc_mcf(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'G&A on Expensed Items',
     1    (ga_exp(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'G&A on Capitalized Items',
     1    (ga_cap(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'Pressure Maint./Cycling',
     1    (inj(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'General O&M',(oam(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'Environmental O&M Costs',(eoam(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'Stimulation Costs',(stim(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'Recompletion Costs',(recomp(iyr),iyr=iyr1,iyr2)
        write(i0,101) 'Intangible Investment',
     1    (ii(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'Intang. Exploratory Costs',
     1      (intang_ewc(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'Intang. Development Costs',
     1      (intang_dwc(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'Base Gas cost             ',
     1      (stor_gas_cost(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'Other Intangible Costs',
     1    (icap(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'Environmental Intangible Capital Costs',
     1    (eicap(iyr),iyr=iyr1,iyr2)
        write(i0,101) 'Portion of Intangibles to Capitalize',
     1    (intcap(iyr),iyr=iyr1,iyr2)
        write(i0,101) 'TOTAL INVESTMENTS'
     1    (ti(iyr)+ii(iyr),iyr=iyr1,iyr2)
        write(i0,101) 'Tangible Investments',
     1    (ti(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'Tang. Exploratory Cost',
     1    (tang_ewc(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'Tang. Development Cost',
     1    (tang_dwc(iyr),iyr=iyr1,iyr2)
        write(i0,102) 'Environmental',
     1    (etcap(iyr),iyr=iyr1,iyr2)
```

```
       write(i0,102) 'Other Tangible Capital',
     1   (otc(iyr),iyr=iyr1,iyr2)
       write(i0,112) 'Compressor     Capital',
     1   (comp(iyr),iyr=iyr1,iyr2)
       write(i0,101) 'Depreciable/Capitalized Investments',
     1   (tci(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'Adj. for Federal Tax Credits',
     1   (tciadj(iyr),iyr=iyr1,iyr2)
       write(i0,101) 'Depreciable/Capitalize Base',
     1   (cap_base(iyr),iyr=iyr1,iyr2)
       write(i0,101) 'Depreciation',
     1   (depr(iyr),iyr=iyr1,iyr2)
       write(i0,101) 'Depletable G&G/Lease Costs',
     1   (la(iyr)*plac+gg(iyr)*pggc,iyr=iyr1,iyr2)
       write(i0,102) 'Lease Acq. Cost',
     1   (la(iyr)*plac,iyr=iyr1,iyr2)
       write(i0,102) 'G&G Costs',
     1   (gg(iyr)*pggc,iyr=iyr1,iyr2)
       write(i0,102) 'Adjustments for Federal Tax Credits',
     1   (dep_crd(iyr),iyr=iyr1,iyr2)
       write(i0,101) 'Depletion Base',
     1   (dggla(iyr),iyr=iyr1,iyr2)
       write(i0,101) 'Expensed G&G/Lease Costs',
     1   (eggla(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'Lease Purchase Cost',
     1   (la(iyr)*(1-plac),iyr=iyr1,iyr2)
       write(i0,102) 'G&G Costs',
     1   (gg(iyr)*(1-pggc),iyr=iyr1,iyr2)
       write(i0,101) 'Net Revenues',
     1   (netsales(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'Operator Severance Taxes',
     1   (sevtax(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'Operating Costs',
     1   (toc(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'Expensed Int.,G&G, and Lease Acq.',
     1   (ii(iyr)-intcap(iyr)+eggla(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'Depreciation',
     1   (depr(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'Depletion Allowance',
     1   (deplet(iyr),iyr=iyr1,iyr2)
       write(i0,101) 'Taxable Income',
     1   (nibta(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'Tax Credit Addback',
     1   (eortca(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'Intangible Addback',
     1   (intadd(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'G&G/Lease Addback',
     1   (ggla(iyr),iyr=iyr1,iyr2)
       write(i0,101) 'Net Income Before Taxes',
     1   (nibt(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'State Income Taxes',
     1   (sttax(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'Federal Income Tax',
     1   (fedtax(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'Federal Tax Credits',
     1   (fedtaxc(iyr),iyr=iyr1,iyr2)
       write(i0,101) 'Net Income After Taxes',
     1   (niat(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'plus Depreciation',
     1   (depr(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'plus Depletion',
     1   (deplet(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'less Depletable Items',
     1   (dggla(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'less Depreciable/Capitalized Items',
     1   (intcap(iyr)+ti(iyr),iyr=iyr1,iyr2)
       write(i0,102) 'less Tax Credit on Expensable Items',
     1   (eortca(iyr)+intadd(iyr)+ggla(iyr),iyr=iyr1,iyr2)
       write(i0,101) 'Annual After Tax Cash Flow',
     1   (aatcf(iyr),iyr=iyr1,iyr2)
       write(i0,101) 'Discounted After Tax Cash Flow',
1   (datcf(iyr),iyr=iyr1,iyr2)
  write(i0,101) 'Cumulative Discounted After Tax Cash Flow',
1   (catcf(iyr),iyr=iyr1,iyr2)
```

Step 10: Loop for pages is closed.

```
        enddo
```

Step 11: Formats for printing out cash flow are declared.

```
   1011 format(t40,a,t50,20(6x,i2,2x))
   1013 format(t1,a,t50,20(1x,f8.3,1x))
   101  format(t1,a,t50,20(1x,f8.2,1x))
   102  format(t2,a,t50,20(1x,f8.2,1x))
   112  format(t2,a,t50,20(1x,f8.4,1x))
   103  format(t3,a,t50,20(1x,f8.0,1x))
   104  format(t4,a,t50,20(1x,f8.0,1x))
   105  format(t5,a,t50,20(1x,f8.0,1x))
   2000 format(t50,20(1x,a,1x))
   return
   end
```

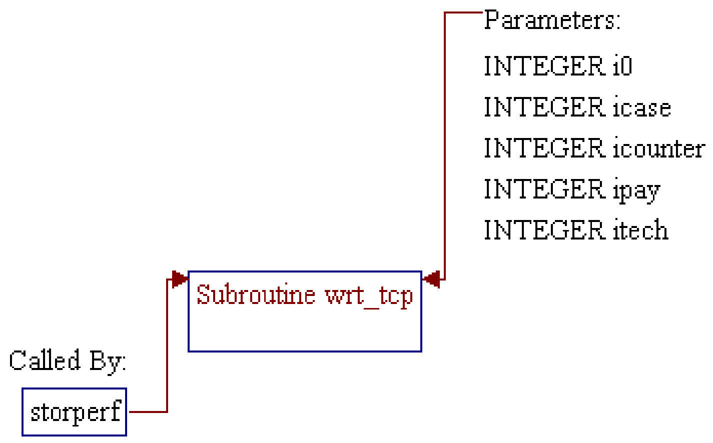
SUB-PROGRAM WRT_TCP()

MAIN THEME: This routine writes out summary of rates, cumulative production, and pressures generated in type-curve routines to output file *.PRD.

READS: None

CREATES: *.PRD

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutines:

• *i0*	Unit number for output file *.PRO (unit 504)
• *maxtim*	Number of time steps.

```
      subroutine wrt_tcp(i0,maxtim)
```

Note: Include files.

```
      include 'dimen.h'
      include 'global.h'
      include 'field.h'
      include 'cost.h'
      include 'tech.h'
      include 'costing.h'
      include 'welldata.h'
      include 'type_out.h'
      include 'gsamvar.h'
      include 'storlp.h'
      include 'rd_data.h'
      include 'type5.h'
```

Step 2: **Profiles of flow rates, cumulative production, and pressures are printed.**

Note: Profiles of four parameters are printed. These parameters are bottom hole pressure (*prbh*), wellhead pressure (*prwh*), gas production (*qrate*), and cumulative production (*cumpay*).

```
      nyr = maxtim
      write(i0,300)
     &  gsamid,fld,
     &  'PBHP',maxtim,
     & (prbh(2,1,iyr),iyr=1,nyr)
      write(i0,300)
     &  gsamid,fld,
     &  'PWHP',maxtim,
     &  (prwh(2,1,iyr),iyr=1,nyr)
      write(i0,302)
     &  gsamid,fld,
     &  'GASP',maxtim,
     &  (qrate(iyr),iyr=1,nyr)
      write(i0,302)
     &  gsamid,fld,
     &  'CUMG',maxtim,
     &  (cumpay(iyr),iyr=1,nyr)

  300 format(a,3x,a,1x,a,1x,i3,1x,60(f8.0,1x))
  301 format(a,3x,a,1x,a,1x,i3,1x,60(f8.0,1x))
  302 format(a,3x,a,1x,a,1x,i3,1x,60(f8.0,1x))
      return
      end
```

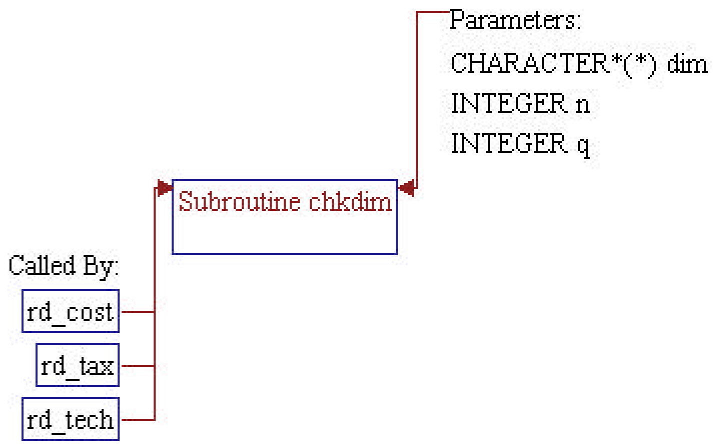
SUB-PROGRAM CHKDIM()

MAIN THEME: Subroutine to check that a dimension has not been exceeded

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: Subroutine declarations and definitions.

Note: Parameters of the subroutine:

• *n*	Number of data
• *q*	Size of array
• *dim*	Name of array

```
      subroutine chkdim(n,q,dim)
```

Step 2: Dimension of array is checked.

```
      integer n,q
      character*(*) dim
      if(n .gt. q) then
        write(6,*) dim,' exceeded - Program must be Recompiled'
        stop
      endif
      end
```

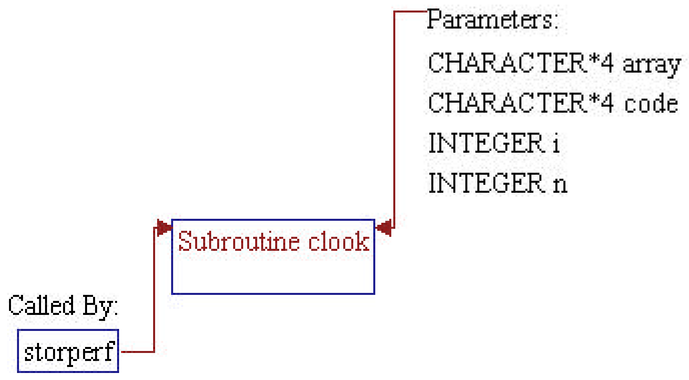
SUB-PROGRAM CLOOK()

MAIN THEME: This routine sequentially searches location of a 4-digit code in a set of string array.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *code*	4-character string
• *array*	Array of string (each entry is 4-character in size)
• *n*	Number of data in array
• *i*	Location of *code* in *array*

```
      subroutine clook(code,array,n,i)
```

Note: Local variables.

```
      integer n,i
      character*4 code,array(*)
```

Step 2: **Location of *code* in *array* is searched.**

```
      do i=1,n
       if(code.eq.array(i)) return
      enddo
      i=0
      return
      end
```

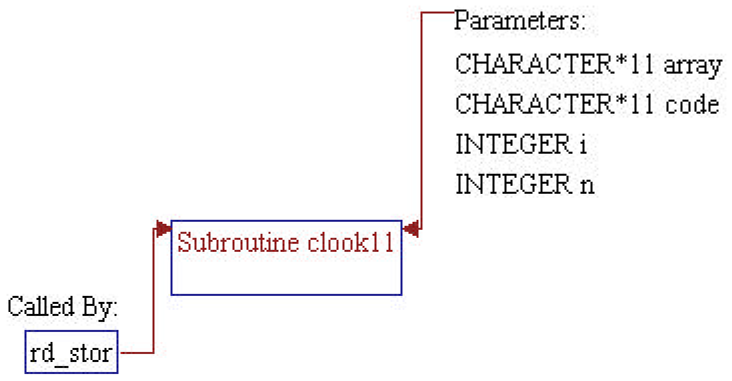
SUB-PROGRAM CLOOK11()

MAIN THEME: This routine sequentially searches location of a 11-digit code in a set of string array.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *code*	11-character string
• *array*	Array of string (each entry is 11-character in size)
• *n*	Number of data in array
• *i*	Location of *code* in *array*

```
      subroutine clook11(code,array,n,i)
```

Note: Local variables.

```
      integer n,i
      character*11 code,array(*)
```

Step 2: **Location of *code* in *array* is searched.**

```
      do i=1,n
       if(code.eq.array(i)) return
      enddo
      i=0
      return
      end
```

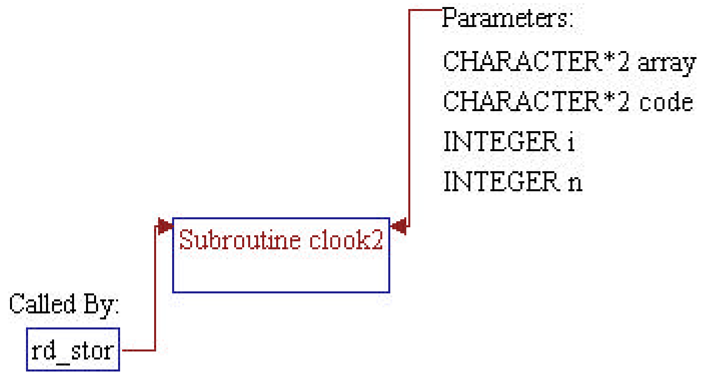
SUB-PROGRAM CLOOK2()

MAIN THEME: This routine sequentially searches location of a 2-digit code in a set of string array.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *code*	2-character string
• *array*	Array of string (each entry is 2-character in size)
• *n*	Number of data in array
• *i*	Location of *code* in *array*

```
      subroutine clook2(code,array,n,i)
```

Note: Local variables.

```
      integer n,i
      character*2 code,array(*)
```

Step 2: **Location of *code* in *array* is searched.**

```
      do i=1,n
       if(code.eq.array(i)) return
      enddo
      i=0
      return
      end
```

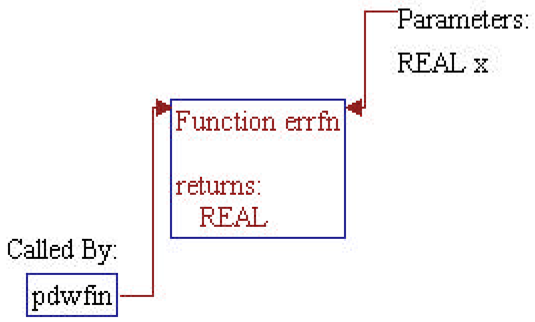
SUB-PROGRAM ERRFN()

MAIN THEME: This function computes the error function based on a polynomial approximation from Abramowitz, M. and Stegun, i.a., handbook of mathematical functions with formulas, graphs and mathematical tables, national bureau of standards applied mathematics series 55, June, 1964 (10th printing dec., 1972, with corrections).

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: Subroutine declarations and definitions.

Note: Parameter of the subroutine:

- *x* real argument

```
      function  errfn(x)
```

Note: Local variables and data.

```
      double precision a1, a2, a3, a4, a5, p, t, f
      data a1,a2,a3,a4,a5,p/
     +   0.254829592d0, -0.284496736d0, 1.421413741d0, -1.453152027d0,
     +   1.061405429d0,  0.3275911d0/
```

Step 2: Error function is calculated.

```
      z = abs(x)
```

Note: first, check that -5<x<5. if x is outside the allowable range, return errfn = +1 or -1

```
      if (z .gt. 5.) then
              errfn = 1.
```

Note: if -0.1<x<0.1, then use the taylor series expansion

```
      else if (z .lt. 0.1) then
              errfn = 1.1283792 * (z - z**3/3. + z**5/10.)
```

Note: otherwise, compute the error function using the abramowitz and stegun approximation.

```
      else
              t = 1.d0 / (1.d0 + p * z)
              f = 1.d0 - t * (a1 + t * (a2 + t * (a3 + t * (a4 +
     +                t * a5)))) * exp (-z**2)
              errfn = real(f)
      end if
      if (x .lt. 0.) errfn = -errfn
      return
      end
```

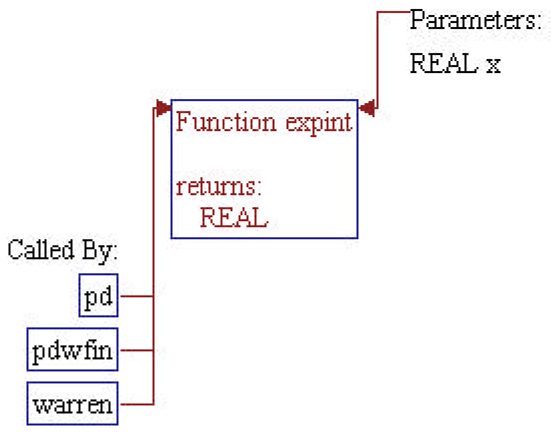
SUB-PROGRAM EXPINT()

MAIN THEME: This function computes the exponential integral function based on polynomial approximations from Abramowitz, M. and Stegun, i.a., handbook of mathematical functions with formulas, graphs and mathematical tables, national bureau of standards applied mathematics series 55, June, 1964 (10th printing dec., 1972, with corrections).

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameter of the subroutine:

- *x* positive, real argument

```
      function   expint (x)
```

Note: Local variables and data.

```
      double precision a0,a1,a2,a3,a4,a5,b1,b2,b3,b4,c1,c2,c3,c4,z
      data a0,a1,a2,a3,a4,a5,b1,b2,b3,b4,c1,c2,c3,c4/
     +   -0.57721566d0, 0.99999193d0,  -0.24991055d0,   0.05519968d0,
     +   -0.976004d-2,  0.107857d-2,
     +   8.5733287401d0,1.8059016973d1, 8.6347608925d0, 0.2677737343d0,
     +   9.5733223454d0,2.56329561486d1,2.10996530827d1,3.9584969228d0/
```

Step 2: **Exponential integral is calculated.**

Note: First, check that 0<x<90.1. if x is outside the allowable range, return expint=0.

```
      if ((x.le.0.) .or. (x.ge.90.1)) then
              expint = 0.
              return
      end if
```

Note: compute the exponential integral using the abramowitz and stegun approximations.

```
      z = dble(x)
      if (x .le. 1.) then
              expint = (a0+z*(a1+z*(a2+z*(a3+z*(a4+z*a5)))) - log(z))
      else
              expint = ((((( z + b1) * z + b2) * z + b3) * z + b4)
     +                 /(((( z + c1) * z + c2) * z + c3) * z + c4)
     +                 / z * exp(-z))
      end if
      return
      end
```

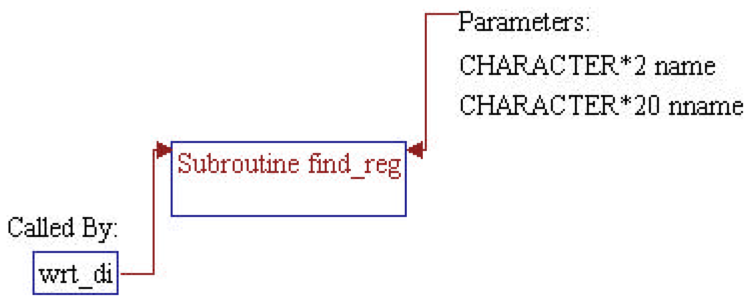
SUB-PROGRAM FIND_REG()

MAIN THEME: This routine converts the region code to region name.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

- *name* Region code
- *nname* Region name

```
      subroutine find_reg(name,nname)
```

Note: Local variables.

```
      character*2  name
      character*20 nname
```

Step 1: **Region name is assigned based on region code.**

```
           if (name .eq. '01') then
             nname = 'New England         '
      else if (name .eq. '02') then
             nname = 'Middle Atlantic     '
      else if (name .eq. '03') then
             nname = 'South Atlantic      '
      else if (name .eq. '04') then
             nname = 'Florida             '
      else if (name .eq. '05') then
             nname = 'East South Central  '
      else if (name .eq. '06') then
             nname = 'East North Central  '
      else if (name .eq. '07') then
             nname = 'West South Central  '
      else if (name .eq. '08') then
             nname = 'West North Central  '
      else if (name .eq. '09') then
             nname = 'Mountain 1          '
      else if (name .eq. '10') then
             nname = 'Mountain 2          '
      else if (name .eq. '11') then
             nname = 'California          '
      else if (name .eq. '12') then
             nname = 'Pacific Northwest   '
      else if (name .eq. '13') then
             nname = 'Canada-East         '
      else if (name .eq. '14') then
             nname = 'Canada-West         '
      else
        write(*,*)'Incorrect storage region,', name
        write(*,*)'Program stopped'
        stop
      end if
      return
      end
```

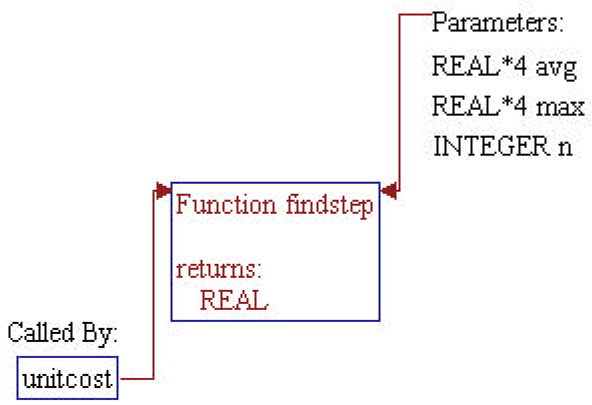
SUB-PROGRAM FINDSTEP

MAIN THEME: Function to determine the location of value *avg* within to entries of array *max*.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *avg*	Real value
• *max*	Real array
• *n*	Size of array *max*

```
      integer function findstep(avg,max,n)
```

Note: Local variables.

```
      real*4 avg,max(*)
      integer n,i
```

Step 1: **Location of *avg* in *max* is searched.**

```
      if(avg.ge.0 .and. avg.le. max(1)) then
        findstep=1
        return
      else
        do i=2,n
          if(avg.ge.max(i-1) .and. avg.le. max(i)) then
            findstep=i
            return
          endif
        enddo
        findstep=n
        return
      endif
      return
      end
```

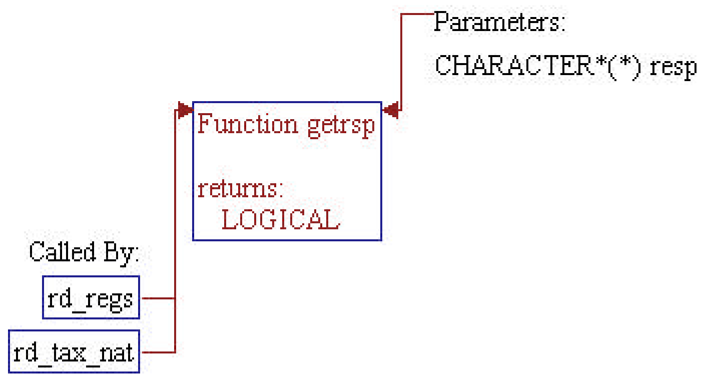
SUB-PROGRAM GETRSP

MAIN THEME: Function that returns logical true if response is yes/YES

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

- *resp* YES or NO response (character)

```
logical function getrsp(resp)
```

Note: Local variable.

```
character*(*) resp
```

Step 1: **Character response (YES/NO) is converted to logical true or flase.**

```
getrsp=.false.
if(index(resp,'Y').gt.0  .or. index(resp,'y').gt.0) then
  getrsp=.true.
elseif(index(resp,'N').eq.0 .and. index(resp,'n').eq.0) then
 write(6,*) resp,' not valid answer to YES/NO'
 stop
endif
return
end
```

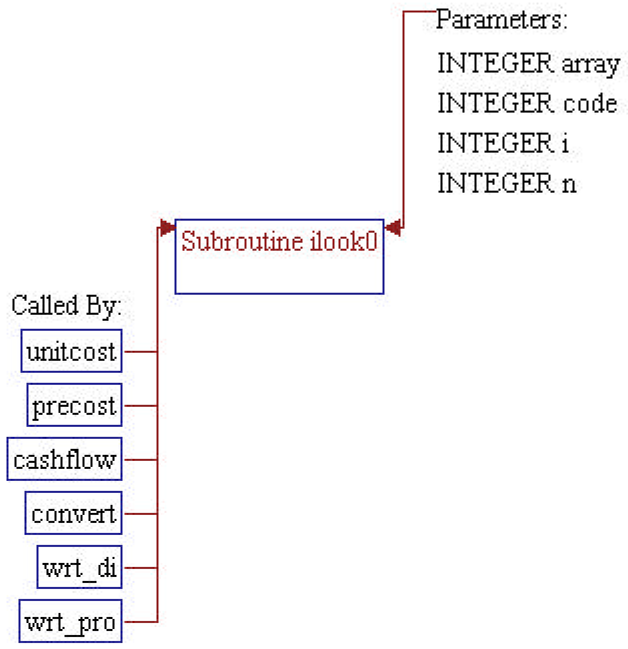
SUB-PROGRAM ILOOKO()

MAIN THEME: This routine sequentially searches location of an integer code in a set of integer array.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *code*	integer code
• *array*	Array of integers
• *n*	Number of data in array
• *i*	Location of *code* in *array*

```
      subroutine ilook0(code,array,n,i)
```

Note: Local variables.

```
      integer n,i
      integer code,array(*)
```

Step 2: **Location of *code* in *array* is searched.**

```
      do i=1,n
       if(code.eq.array(i)) return
      enddo
      i=0
      return
      end
```

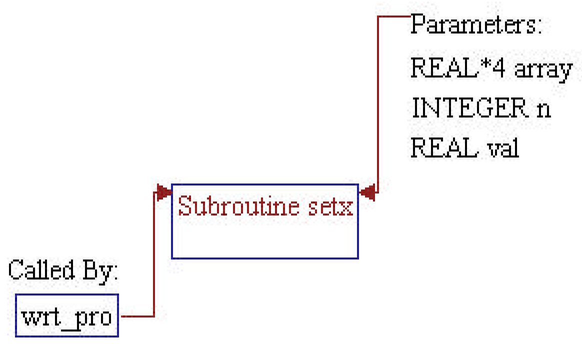
SUB-PROGRAM SETX()

MAIN THEME: The purpose of this routine is to initialize a real type array with the value *val*.

READS: None

CREATES: None

ROUTINE INTERACTIONS:



Step 1: **Subroutine declarations and definitions.**

Note: Parameters of the subroutine:

• *array*	Array of real values
• *n*	Number of data in array
• *val*	Real value

```
      subroutine setx(array,n,val)
```

Note: Local variable.

```
      real*4 array(*)
```

Step 2: **Entries of array *array* is set to *0*.**

```
      do 1 i=1,n
      array(i)=val
    1 continue
      return
      end
```

PROGRAMMER'S GUIDE FOR THE EXPLORATION AND PRODUCTION (E&P) MODULE OF THE GAS SYSTEMS ANALYSIS MODEL (GSAM)

FINAL REPORT

Volume IIIc –E&P Programmer's Guide

For:

U.S. Department of Energy
National Energy Technology Laboratory
Morgantown, West Virginia
Under Contract Number: DE-AC21-92MC28138

By:

ICF Consulting, Inc.
Fairfax, Virginia

February 2001

E&P PROGRAMMER'S GUIDE GENERAL SETUP

This document provides a detailed explanation of all the major subroutines in the Exploration and Production (E&P) module of GSAM. In the next few pages the basic structure of the E&P module is explained, followed by an explanation of the structure of this document and finally a discussion of the explanation of the subroutines within the document.

The Three E&P Executables

The E&P module has three main executables which perform primary E&P functions. Other subroutines are called from these main programs or use the output from these main programs. The three main programs are ENV_WRTE.EXE, MAKEBIN.EXE and EXPLPROD.EXE.

ENV_WRTE.EXE: This routine reads in environmental costs by state from the GSAM environmental module and raw environmental data from data files (*.ENV files such as GSAM1.ENV, GSAM2.ENV, etc.) created from the Reservoir Performance Module and then writes them by GSAMID. In addition, it also creates the gas processing cost file which contains costs for gas processing/treatment. This routine needs to be run before the rest of the E&P module.

MAKEBIN.EXE: This program creates the data bank files for both discovered and undiscovered reservoirs (i.e., UNDB.BNK, UNDB.TCP, for undiscovered reservoirs and DISB.BNK, DISB.TCP for discovered reservoirs). This routine also needs to be run before the rest of the E&P module.

EXPLPROD.EXE: This routine is the main control center for the GSAM E&P Module. It calls other routines which:

a) reads in various input parameters to set up an E&P run (i.e., EXDVI1)
b) reads in the various variables which are stored in the binary files produced as an output of the Reservoir Performance module (i.e., EXDVI2)
c) reads in the environmental and processing cost data (i.e., ENV_READ)
d) calculates the percentage of the total production for undiscovered reservoirs on which any applicable royalty incentive would be available (i.e., EXDVI4A)
e) decides which exploration and development options will be selected each year (i.e., EXDVST)
f) makes reservoir shut-in decisions and calculates the output variables (i.e., EXDVSO)

The general setup of the EXPLPROD.EXE executable is shown in the following flowchart. This flowchart shows the general flow of data and variable in the code. It also shows various input files that lead to the model.

The General Structure of the Document

This document is coherently structured with important routines (over and above the three main routines discussed above) separated by labeled tabs. The write up within each tab contains the main routine (for which the tab is specified) and may also contain other subroutines which it calls. To assist in locating the different subroutines (in case there is more than one within a tab), a table of contents has been provided in each tab. The tabs have been made for the following FORTRAN programs and appear in the document in the following sequence.

EXPLPROD.FOR	
EXDVI1.FOR	
EXDVI2.FOR	
ENV_READ.FOR	
EXDVI4A.FOR	
EXDVI4.FOR	
EXDVST.FOR	(This also contains PKNUDS.FOR, PRJSRT.FOR)
DVL_MSP.FOR	(This also contains GETT.FOR)
EXDVSO.FOR	
ENV_WRTE.FOR	(This also contains PROCESS, BY_REV, CALC_UCC, CAP_COST, CHECK1, CHECK2, CHECK3, CLAUSCC, CLAUSOC, CRYOCC, CRYOOC, DEATRTCC, DEATRTOC, DIRECTCC, DIRECTOC, OP_COST, STRTCC, STRTOC, WT_CST, GLYCOLCC, GLYCOLOC, NITROCC, NITROOC subroutines)
MAKEBIN.FOR	(This also contains EXDVI3.FOR)

The Structure of the Explanations

Within each subroutine the explanations are very detailed explaining pertinent sections of code. The standard format followed for the explanations in each subroutine is as follows:

a) Before the explanations for the code begin there are five subheadings
   i) CALLED BY: Here the other subroutines, that call the subroutine in question, are listed with their brief description.
   ii) CALLS: Here the other subroutines, that the subroutine in question calls, are listed with their brief description.
   iii) READS: Here the input files read in by the subroutine in question are listed with their brief description.
   iv) CREATES: Here the output files created by the subroutine in question are listed with their brief description.
   v) MAIN THEME: This comprises a brief synopsis of the subroutine in question.

b) These five headings may not all appear in each subroutine. For example, if a subroutine does not create any output files, there will not be any subheading ‘CREATES:’.
c) These subheadings are followed by detailed explanations for the code. Most of the code is explained in steps, i.e., the explanation for a chunk of related code is delegated to a single step. Between steps if a certain section of code needs an explanation a ‘Note’ is inserted with the relevant explanation.

Table of Contents

SUBROUTINE EXPLPROD.FOR

CALLS: GETT (Estimates the time required for each phase of the subroutine.)
EXDVI1 (Reads in various input parameters to set up an E&P run.)
EXDVI2 (Reads in the various variables which are stored in the binary files produced as an output of the Reservoir Performance module.)
ENV_READ (Reads in the environmental and processing cost data.)
EXDVI4A (Calculates the percentage of the total production for undiscovered reservoirs on which any applicable royalty incentive would be available.)
EXDVST (Decides which exploration and development options will be selected each year.)
EXDVSO (The reservoir shut-in decisions are made and the output variables are calculated.)

MAIN ROUTINE: This routine controls the main flow through the GSAM exploration/development model

Step 1: The time variables are initialized.

```
      do 10 i=1,15
      tmes(i)=0.0
      tmea(i)=0.0
10    continue
      call gett(tmes(1),tmea(1),0)
```

Step 2a: The input specifications are read.

```
      call gett(tmes(2),tmea(2),0)
      call exdvi1
      call gett(tmes(2),tmea(2),1)
      call gett(tmes(3),tmea(3),0)
      call exdvi2
      call gett(tmes(3),tmea(3),1)
```

Step 2b: The environmental specifications are read.

Note: The environmental information and costs are read for various environmental cases.

```
      call env_read
```

Step 2c: The fractional NPV ($) of the portion of production on marginal royalty production, and the fraction of NPV on which the incentive is applicable, to total NPV (fraction) are calculted.

```
      call exdvi4a
```

Step 3: Drilling and production actions for each year are determined.

```
      call gett(tmes(4),tmea(4),0)
      call exdvst
      call gett(tmes(4),tmea(4),1)
```

Step 4: The summary report is printed.

```
    call gett(tmes(5),tmea(5),0)
    call exdvso
    call gett(tmes(5),tmea(5),1)
```

Step 5: The summary of time required to process each step is printed.

```
      call gett(tmes(1),tmea(1),1)

      do 5210 i=1,15
      tmef(i)=tmea(i)/tmea(1)
5210  continue
      tmef(6)=tmea(6)/ tmea(3)
      tmef(7)=tmea(7)/ tmea(3)
      tmef(8)=tmea(8)/ tmea(4)
      tmef(9)=tmea(9)/ tmea(4)
      tmef(10)=tmea(10)/ tmea(4)
      do 5220 i=1,15
      tmea(i)=tmea(i)/(100.0*60.0)
      write(*,5211) i,tmea(i),tmef(i)
5211  format(' tme (minutes/share): ',i2,2f10.4)
5220  continue
```

Step 6: The program ends and the output files are closed.

```
close(46)
close(31)

stop
end
```

SUBROUTINE EXDVI1

CALLED BY:	EXPLPROD (Controls the main flow through the expl./dvlp. model.)
CALLS:	ERRMSG (Prints out errors and warnings) CLOOK20 (Compares two variables from different sources)
READS:	‘gen_tml.spc’ (Contains various inputs: Prices, Royalty rate, etc.) ‘node.spc’ (Contains supply region specifications for plays.) ‘drl_cst.spc’ (Contains entries to calculate exploratory drilling well costs.) ‘sup_cse.dat’ (Specifies supply case.) ‘gasprc.new’ (Contains gas price forecasts.) ‘drl_rcp.spc’ (Contains drilling rig capacity specifications.) ‘drl_cap.spc’ (Contains various drilling specifications.) ‘ply_dfn.spc’ (Contains play level specifications.) ‘etec_pen.spc’ (Contains exploration technology penetration rates.) ‘exp_dfn.spc’ (Contains exploration definition specifications.) ‘exp_ply.spc’ (Contains detailed specifications for exploration risk.) ‘resav.spc’ (Contains reserve growth data.) ‘resavrg.spc’ (Contains reserve growth data.) ‘dtec_pen.spc’ (Contains development technology penetration rates.) ‘dtec_fed.spc’ (Contains dev. technology penetration rates for federal lands.) ‘etec_fed.spc’ (Contains expl. technology penetration rates for federal lands.) ‘tax_cde.spc’ (Contains the tax specifications.) ‘tax_det.spc’ (Contains the tax specifications. Not currently used.) ‘exp_cst.spc’ (Not currently used.) ‘dvl_tpr.spc’ (Defines development technology parameters.)
MAIN THEME:	This program reads in various input parameters to set up an E&P run. The routine performs internal consistency checks (in-case of errors it prints error messages on the screen). The data also gets processed and converted into useable variables before being passed over to other routines of the E&P module. The general format followed while reading a specific file in this subroutine follows the following sequence: a) The data is read. b) Internal consistency checks are performed to make sure that the data

is consistent, if it isn't error messages are printed.
c) The data is processed and stored in an acceptable format so that it can be transferred to other parts of the E&P module.

Step 1: **The input file 'gen_tml.spc' is opened, the data is read and the file is closed.**

Note: This file contains years which specify the time periods (tme) for which the has price (in file gasprc.new) is available. It also contains the discount rate (disrte), the screening price (not used currently). For missing years the gas price is determined interpolating the gas price for which there is data, the start year for the model (tmex), the marginal gas rate which classifies all wells with a gas production less than this rate as being marginal gas wells (rate_marg), the royalty rate applicable on federal lands (%) (roy_incentive), and the variable 'iwrt_play' which whether the prodsumm.out file contains entries by GSAM region (in this case iwrt_play is set to 0) or by play specific details in addition to the regional output (in this case iwrt_play is set to 1).

```
        open(11,file='gen_tml.spc')

        read(11,101) (tme(t),t=1,mxntme)
        read(11,*) disrte
        read(11,*) scrprc
        read(11,*) tmex
        read(11,*) rate_marg
        read(11,*) roy_incentive
        roy_incentive = roy_incentive/100.0
        read(11,*) iwrt_play
        iocde = 2

101     format(33(i4,1x))
        close(11)
```

Step 2: **The number of time periods (ntme) is determined from the maximum number of time periods (mxntme).**

Note: It is confirmed that the specified years are correctly ordered. If they are not in order then an error message is printed.

```
        ntme=mxntme
        do 105 t=1,mxntme
        if(tme(t).ne.0) then
        if((t.gt.1).and.(tme(t).le.tme(t-1))) then
        call errmsg(4,101)
        endif

        else
        ntme=min0(ntme,t-1)
        endif
        105 continue
        if(ntme.le.0) call errmsg(4,101)
```

Step 3: **The number of years (nyr) in the current scenario is estimated.**

Note: This is done by subtracting from the last calendar year for which data exists (i.e.tme(ntme)) the first year (i.e. (tmex)) from which the model starts.

```
        nyr=tme(ntme)-tmex+1
```

Step 4: **The supply region specifications is read from 'node.spc' & 'drl_cst.spc'.**

Note: The input files are opened, the the supply region specifications initialized. Specifically the number of supply regions (nsrg) is set to zero, name of the supply regions (srgnme). The code is used to indicate if the supply region is processed or not. The variable ‘srge’ is set to zero if the supply region is not processed yet and is set to one when the supply region gets processed.

```
        open(12,file='node.spc')
        open(13,file='drl_cst.spc')
        nsrg=0
        do 110 s=1,mxnsrg
        srgnme(s)='                    '
        srgc(s)=0
110     continue
```

Note: The node name (nname) and the supply region number (s) are read. Nodes in GSAM can be either supply nodes, demand nodes or LNG nodes. These supply nodes have a unique non-zero region counter/number.

```
120     read(12,121,end=130) nname,s
121     format(a20,2(i2,1x),f6.3,1x,f6.3)
```

Note: If the region read is a supply region (i.e. it has a unique non-zero supply region number) then it is first confirmed that the region number has not already been used and that it is within bounds (i.e. less than the maximum number of supply regions available i.e., mxnsrg) and then appropriately saved. This section of the code also reads the drilling cost specifications, namely, the various coefficients in the fourth order drilling cost equation (drlincst variables) and the supply region identifier (si).

```
        if(s.gt.0) then
        if(s.gt.mxnsrg) call errmsg(4,104)
        if(srgc(s).ne.0) call errmsg(4,105)
        srgnme(s)=nname
        srgc(s)=1
        nsrg=max0(nsrg,s)
        read(13,*,end=130)si,drlincst(4,si),drlincst(1,si),
        * drlincst(2,si),
        * drlincst(3,si),drlincst(5,si)
        if(si.ne.s)call errmsg(3,121)
        endif
        go to 120
```

Note: The file is closed and it is confirmed that the supply region specifications include at least one region and there is no missing data. If no region is specified in the node.spc file (i.e. srgc = 0) then a fatal error message is printed and the program terminated.

```
130     close(12)
        close(13)
        if(nsrg.eq.0) call errmsg(4,108)
        do 135 s=1,nsrg
        if(srgc(s).eq.0) call errmsg(4,110)
135     continue
```

Step 5: **The supply price case to run is read from 'sup_cse.dat'.**

Note: This variable (iscase) is used to pick the correct supply price track from the gasprc.new file.

```
        open(13,file='sup_cse.dat')
        read(13,151) iscase
151     format(i2)
        close(13)
```

Step 6: **The supply price specifications are read in from 'gasprc.new' by time periods specified in gen_tml.spc file.**

Note: The gasprc.new input file is opened.

```
open(14,file='gasprc.new')
```

Note: The price specifications for all cases up to and including the one specified for this scenario are read by looping through all the supply cases. The input for the specified price case overwrites the input of the previous cases. The inputs read include the supply region name (nname), two temporary indices (nsps,jj) which are no longer used (could be used for supply region-state crosswalks), and the well-head supply prices (supnpr in $/MCF) for the time periods as specified in the gen_tml.spc file.

```
        do 200 ii=1,iscase
        if(ii.le.iscase) then
        do 190 s=1,nsrg
        read(14,152,end=195) nname,nsps,jj,(supnpr(t,s),t=1,ntme)
152     format(a20,2i3,33(1x,f7.3))
        if(nname.ne.srgnme(s)) call errmsg(4,401)
190     continue
        endif
        go to 200
195     call errmsg(4,407)
200     continue
```

Note: The input file is closed.

```
        close(14)
```

Step 7: **Time period prices are converted to yearly prices.**

Note: The following piece of code specifies loops so that prices for each region (s) and year (y) are calculated for every year in the following steps.

```
        do 250 s=1,nsrg
        do 240 y=1,mxnyr
```

Note: The calendar year (year i.e., 1995, 1996 ….) is calculated. The calendar year (year) variable is then used to get identifiers for interpolation to be done in the next step.

```
        year=tmex+y-1
        do 210 t=1,ntme
        if(year.le.tme(t)) go to 215
210     continue
        t=ntme
```

Note: The annual price (supprc) is calculated by interpolating linearly between years, using a temporary variable (vscl) and the input well-head supply prices (supnpr) for the time periods as specified in the gen_tml.spc file.

```
215     if(t.gt.1) then
        vscl=float(tme(t)-year)/float(tme(t)-tme(t-1))
        vscl=amin1(1.0,amax1(0.0,vscl))
        supprc(y,s)=vscl*supnpr(t-1,s)+(1.0-vscl)*supnpr(t,s)
        else
        supprc(y,s)=supnpr(1,s)
        endif
240     continue
250     continue
```

Step 8: **The existing drilling footage capacity specifications are read in from 'drl_rcp.spc'.**

Note: Here the input file 'drl_rcp.spc' is opened and the capacity for the supply regions for both exploration (drlrcp(s,1)) and development drilling (drlrcp(s,2)) is initialized. In addition, sgrc(s), the counter for indicating whether the supply region has been processed or not is initialized.

```
        open(15,file='drl_rcp.spc')
        do 311 s=1,nsrg
        drlrcp(s,1)=0.0
        drlrcp(s,2)=0.0
        srgc(s)=0
311     continue
```

Note: The input specifications are read until the end of the file is reached. The various inputs read include the supply region name (nname), starting drilling rig capacity (in thousand ft) for exploration (vale), the starting drilling rig capacity (in thousand ft) for development (vald), the maximum net increase in footage in a year (chgreg), the cost to move a rig into the region, per well (in thousand's of dollars) (chgrigs), the rig movement factor (cap_move) a value of "0" means one can not move rig capacity out of a region.

```
323     read(15,321,end=351) nname,vale,vald,chgreg,chgrigs,cap_move
321     format(a20,2f7.1,t40,f9.0,t50,f9.0,t60,f9.0)
```

Note: Here the input data is matched to a supply region and the input specifications are saved, (for example, srgnme(s) as nname, dr_cap(s) as cap_move, dr_rig(s) as chgrigs/100.0, dr_reg(s) as chgreg/100.0, drlrcp(s,1) as vale*1000.0, drlrcp(s,2) as vald*1000.0). Then the region is marked as having “data read” (srgc is set to one) and the next input record is processed. This continues till there are no more regions to process. If there is a problem in the procedure an error message is printed on the screen.

```
        do 331 s=1,nsrg
        if(nname.ne.srgnme(s)) go to 331
        if(srgc(s).ne.0) then
        write(*,322) nname
322     format(' Supply Region: ',a20)
        call errmsg(4,402)
        endif
        dr_cap(s) = cap_move
        dr_rig(s) = chgrigs/100.0
        dr_reg(s) = chgreg/100.0
        drlrcp(s,1)=vale*1000.0
        drlrcp(s,2)=vald*1000.0
        srgc(s)=1
        go to 323
331     continue
        write(*,322) nname
        call errmsg(4,403)
```

Note: The input file is closed and it is confirmed that data for all regions is read in.

```
351     close(15)
        do 361 s=1,nsrg
        if(srgc(s).eq.0) then
        write(*,322) srgnme(s)
        call errmsg(4,404)
        endif
361     continue
```

Step 9: **The drilling capacity specifications are read in from 'drl_cap.spc'.**

Note: The following variables are read.

a) Drilling Efficiency (drleff) - Drilling efficiency is defined in GSAM's E&P module as the percentage gain in efficiency when rigs move from exploration to development drilling (or loss in vice versa). For example: 1.2 means that when rigs move from exploration to development, their drilling capacity increases by 20% and conversely when rigs move from development to exploration their capacity decreases by 17% (1 - 1/1.2).
b) Rig Retirement Rate (drlrrt) - The rig retrirement rate is the percent of drilling capacity (or rigs) that get retired every year.
c) Variable (drlvcs) and Full Drilling Costs factors (drlfcs) ($ Thousand/ft)
d) Rig Utilization Rate (%) at Which Variable Drilling Cost Begin -

The regional rig utilization rate drives the drilling cost as follows. As rig utilization, as a percent, increases, the costs of using rigs also increases because of the increased demand for rigs, up to the full drilling cost.

The current values in 'drl_cap.spc' of the minimum variable rig utilization rate and the rig utilization rate at which the variable drilling cost begins are 40% and 70% respectively. When the regional rig utilization rate is less than 40%, low demand for rigs drives down the drilling costs. The drilling costs, calculated for development wells in the Reservoir Performance Module and for exploration wells in the E&P Module are subjected only to variable drilling costs and are therefore reduced by a factor. This factor is the ratio of the variable drilling cost factor over the full drilling cost factor: 220/271 = 81%, so that in this case, for regions which have less than 40% rig utilization, the drilling costs are 81% of the full cost. When the regional rig utilization rate is above 70%, full drilling costs prevail, that is, fully 100% of the drilling costs calculated for development wells in the Reservoir Performance Module and for exploration wells in the E&P Module are applied. When the regional rig utilization rate is between 40% and 70%, a linear interpolation between the drilling cost factors is performed (between 81% and 100%) to calculate a drilling cost factor in accordance with the rig utilization rate. The regional rig utilization rate may exceed 100%. This phenomena of "super demand" can occur when rigs are moved into the high-demand region. Because moving drilling rigs is costly, the drilling cost factor will rise above 100% of the full drilling costs. Note that GSAM does

not explicitly model indvidual drilling rigs, rather it models regional drilling capacity in terms of footage drilled.
e) Maximum Change in Rig Fleet (105 means 5% increase) - The maximum change in the rig fleet indicates how much construction of capacity can occur in one year.
f) Annual Reduction in Drilling Costs (%)
g) Minimum number of wells that could be drilled in a reservoir in a year.
h) Maximum percentage of total wells that could be allowed in a year.

```
open(16,file='drl_cap.spc')
read(16,*) drleff
read(16,*) drlrrt
read(16,*) drlvcs
read(16,*) drlfcs
read(16,*) drlchv
read(16,*) drlchf
read(16,*) drlchx
read(16,*) drlcim
read(16,*) xdmnwl
if (xdmnwl.le.0.0) xdmnwl = 1.0
read(16,*) xddfrc
xddfrc = xddfrc/100.0
if (xddfrc.le.0.0) xddfrc = 0.1

drlfcs=drlfcs/100.0
drlvcs=drlvcs/100.0

close(16)
```

Step 10: **The play specifications are read in from the 'ply_dfn.spc' file and are mapped to the supply region specifications.**

Note: The input specifications are then read until the end of file reached. The following variables are read in:

a) Play identifier (4-digit play code) (pname)
b) Region name (nname)
c) State code (currently not used) (istate)
d) First technology (current) development success rate (vscl1)
e) Second technology success rate (vscl2)
f) Third technology (advanced) development success rate (vscl3)
g) Dominant resource type of the play (rsty1)

The dominant resource type as used in this file is used to aggregate reservoirs in specific categories. If a play is predominantly conventional with a few reservoirs being water-drive, then the RP Module uses water-drive type curve to predict the performance of these reservoirs, but in aggregation in the E&P Module all of the reservoirs of the play would be aggregated into the conventional category.

Index Description for Dominant Resource Type.

1	Conventional
2	Tight
3	Associated gas
4	Naturally fractured reservoir with induced massive hydraulic fracture
5	Water-drive
6	Coal/shale
7	Analyzed resource (currently modeled as Gulf offshore reservoirs)

h) Average depth of play (dpth) which is used to compute exploration costs.
i) Royalty Rate (%) (roy)
j) % of play on Federal land (frac)

```
        open(17,file='ply_dfn.spc')
        nply=0

        read(17,*)
        read(17,*)

410     read(17,411,end=450) pname,nname,istate,vscl1,vscl2,vscl3,rsty1,
   *    dpth,roy,frac

411     format(2a20,i2,3f6.1,i2,f9.1,f6.1,f8.2)
```

Note: The following piece of code determines the validity of the state counter. The state counter should not be negative or more than 50.

```
        if((istate.le.0).or.(istate.gt.50)) then
        write(*,441) pname,nname
        call errmsg(4,406)
        endif
```

Note: The play is located in the list of plays and the specifications are saved (pname as plynme(nply), rsty1 as rst(nply), istate as plyste(p) dpth as etcdep(p), vscl1/100.0 as plydsc(1,p), vscl2/100.0 as plydsc(2,p), vscl3/100.0 as plydsc(3,p), roy/100.0 as royrate_f(p) and frac/100.0 as frac_fed(p)).

In actual computation in the E&P module royalty rates and federal fraction of a reservoir are read from the env_stat.spc file.

```
        p=0
        if(nply.gt.0) then
        do 420 p=1,nply
        if(pname.eq.plynme(p)) go to 425
420     continue
        p=0
425     continue
        endif
        if(p.eq.0) then
        nply=nply+1
        if(nply.gt.mxnply) call errmsg(4,405)
        ply4(nply) = pname(1:4)
        plynme(nply)=pname
        rst(nply)=rsty1
        p=nply
        ply4(p) = pname(1:4)
        plyrga(p)=0
        plyste(p)=istate
        etcdep(p)=dpth
        croyrate_f(p) = roy/100.0  ! now read from env_stat.spc file
        cfrac_fed(p) = frac/100.0  ! now read from env_stat.spc file
        do 430 rsty=1,mxrsty
        plydsc(1,p)=vscl1/100.0
        plydsc(2,p)=vscl2/100.0
        plydsc(3,p)=vscl3/100.0
430     continue
        endif
```

Note: The supply region – play name crosswalk is done. If a match is not made on the supply region then a fatal error message is printed.

```
        do 440 s=1,nsrg
        if(nname.ne.srgnme(s)) go to 440
        plyrga(p)=s
        go to 410
440     continue
        write(*,441) pname,nname
441     format(' Play: ',a20,' Supply Region: ',a20)
        call errmsg(4,406)
```

Note: The input file is closed.

```
450     close(17)
```

Step 11: **The undiscovered (plynfl) and discovered/undeveloped fields (i.e., banked, (plybfl) fields) are initialized by field size and play and assigned to the number of field size specifications (‘nfsz’ generally 13, i.e., from 5 through 17) for a play.**

```
        do 510 p=1,nply
        do 505 f=1,mxnfsz
        plynfl(f,p)=0.0
        plybfl(f,p)=0.0
505     continue
510     continue
        do 570 p=1,nply
        nfsz(p)=0
        do 560 f=1,mxnfsz
        if((plynfl(f,p).ne.0.0).or.(plybfl(f,p).ne.0.0)) then
        nfsz(p)=max0(nfsz(p),f)
        endif
560     continue
570     continue
```

Step 12: **The Exploration Technology Penetration Curves are read in from 'etec_pen.spc'.**

Note: The file 'etec_pen.spc' is used to model the penetration curve of current and advanced exploration technology. Exploration technology penetration affects only the decision to apply a given exploration and efficiency to find remaining undiscovered resources in given plays. The file 'etec_pen.spc' can be used to measure the impact of increasing or decreasing an exploration technology's market penetration at any point in time. Note that all of the resource types should be specified for a year. The values are interpolated (or extrapolated) for years in which data is not provided.

Note: Here the number of exploration technologies (netc) is initialized and the input file is opened ('etec_pen.spc').

```
        netc=0
        open(19,file='etec_pen.spc')
```

Note: All the specifications are read and it is confirmed that penetration is no less than 0 and no greater than 100%. The variables read include:

a) Exploration technology parameter name (nname), they are:

Current - Conventional
Current - Tight
Current - Radial Flow
Current - Linear Flow

Current - Water Drive
Current - Unconventional
Current - Analyzed
Advanced - Conventional
Advanced - Tight
Advanced - Radial Flow
Advanced - Linear Flow
Advanced - Water Drive
Advanced - Unconventional
Advanced - Analyzed
Exploration technology penetration rates could be specified for all of the above categories.

b) Time Period (year)

c) Exploration technology penetration rate (%) (vscl)

d) Resource Type (rsty)

```
600     read(19,601,end=650) nname,year,vscl,rsty
601     format(a20,i4,f6.1,i6,2f9.6)
        vscl=amax1(0.0,amin1(vscl,100.0))
```

Note: The exploration technology is searched in the list of technologies. If it is found then it is added to the list (etcnme(z) = nname) and the penetration specifications (etcpen) are initialized to -1. Otherwise an error message is printed.

```
        z=0
610     z=z+1
        if(z.gt.netc) go to 620
        if(nname.ne.etcnme(z)) go to 610
        go to 630
620     netc=netc+1
        if(netc.gt.mxnetc) call errmsg(4,410)
        etcnme(z)=nname
        if((rsty.le.0).or.(rsty.gt.mxrsty)) call errmsg(4,410)
        etcrst(z)=rsty
        do 625 y=1,mxnyr
        etcpen(y,z)=-1.0
625     continue
```

Note: The year (y) is chosen and its validity is confirmed (y should be between one and the mxnyr). It is also verified that the penetration for the technology has not already been specified (etcpen(y,z) not equal to -1.0). The input penetration specifications are then saved (etcpen(y,z) as vscl/100.0).

```
630     y=year-tmex+1
        if((y.le.0).or.(y.gt.mxnyr)) then
        write(*,631) nname,year
        call errmsg(4,411)
        endif
        if(etcpen(y,z).ne.-1.0) then
        write(*,631) nname,year
631     format(' Exp-Technology: ',a20,' Year: ',i4)
        call errmsg(4,412)
        endif
        etcpen(y,z)=vscl/100.0
        go to 600
```

Note: The input file is closed and it is confirmed that some data for the exploration technology exists.

```
650     close(19)
        if(netc.le.0) call errmsg(4,413)
```

Note: The exploration penetration rate data for missing years is computed by interpolation.

```
        do 690 z=1,netc
        y1=0
        y2=0

        do 680 y=1,mxnyr
        if(etcpen(y,z).ne.-1.0) then

        y1=y
        else

        if(y2.lt.y) then
        do 660 y2=y,mxnyr
        if(etcpen(y2,z).ne.-1.0) go to 665
660     continue
        y2=mxnyr+1
665     continue
        endif

        if((y1.eq.0).and.(y2.gt.mxnyr)) then
        write(*,666) etcnme(z)
666     format(' Exp-Technology: ',a20)
        call errmsg(4,414)
        elseif(y1.eq.0) then
        etcpen(y,z)=etcpen(y2,z)
        elseif(y2.gt.mxnyr) then
        etcpen(y,z)=etcpen(y1,z)
```

```
          else
          vscl=float(y2-y)/float(y2-y1)
          etcpen(y,z)=vscl*etcpen(y1,z)+(1.0-vscl)*etcpen(y2,z)
          endif
          endif
680       continue
690       continue
```

Step 13: **The Exploration Discovery Patterns (matrix) are read in by field size class and technology type from 'exp_dfn.spc'.**

Note: The file EXP_DFN.SPC file is used in GSAM to model the uncertainty inherent in exploration practices. The uncertainty, in this case, concerns the probability of finding accumulations (reservoirs) in a specific field size class (FSC) of a play. Modeling this uncertainty is based on the premise that the remaining reservoirs that are undiscovered are of a size less than or equal to a maximum field size class already explored.

Given that a FSC is available, 15 or less in this example, the uncertainty arises in the chances of finding any reservoir; because reservoirs of FSC 15 are available, they are the most likely to be found in this case. The next most likely would be FSC 14, and so on down to FSC 5. According to this logic, a weight must be assigned to a reservoir size, indicating that it is more or less likely to be found, and this is the purpose of the EXP_DFN.SPC file's matrix. Each row in the matrix in EXP_DFN.SPC can be thought of as the exploration curve , in any year, for a given technology, for a given resource type, for a given FSC availability.

Given that a FSC is available to be discovered, the relative weights on the probablility of discovery are derived from one assumption: that the chance of finding a larger reservoir (one with a greater area) is better than the chances of discovering a smaller reservoir. Reservoirs that have a FSC of 10 or below are too small to differentiate them from one another, so that the weights on the chances of finding any of these reservoirs are based only on the area of the reservoirs in a FSC. When the area is the only factor that determines the probability of reservoir discovery, the discovery process is considered "random". Reservoirs having a FSC of 17 to 11 also base their probability weights on area, but include an additional factor that takes into account the ease of differentiation among these larger reservoirs by using technology.

As field size classes increase, the volume in the FSC doubles

(FSC 10 has an average OGIP of 19.2 and FSC 11 has an average OGIP of 38.4) (1). Also, the thickness and area are assumed to be linearly related. From these two pieces of information, the relative probability weights based on area can be derived. If reservoirs of FSC 10 are the largest remaining available, the relative weight on the probability of finding a reservoir of FSC 10 is 1. The relative weight on the probability of finding a reservoir of FSC 9 is less than that of FSC 10 by a multiplying factor. This factor is (1/√2), and is calculated by transforming the following ratio:

Note: Area*Thickness = Volume or Ah = OGIP and Ax is area in FSCx and hx is thickness in FSCx

A9h9 / A10h10 = 1/2 because of (1) above
h = cA from (2) above; where c is constant of proportionality
A9(cA9) / A10(cA10) = 1/2
A92 / A102 = 1/2
A9/ A10 = 1/ 2√ (1/2)

Therefore, if FSC 10 is the largest available, the weight on the chances of finding a reservoir in FSC 10 = 1,
FSC 9 = 1(1/ √ (1/2)) = 0.7071,
FSC 8 = 1(1/ √ (1/2))(1/ √ (1/2)) = 0.50, etc.
For reservoirs in a FSC above 10, the same formula applies, with an additional factor. The model assumes that for these larger reservoirs, the chances of finding a reservoir in a FSC is 25% better than random for each larger FSC in conventional, water-drive, and offhshore reservoirs and 75% better than random for each larger FSC in tight, radial and linear flow, and unconventional. So that if FSC 15 is the largest size of the remaining conventional reservoirs, the chance of finding a FSC 15 reservoir is assigned a weight of 1.
For FSC 14 the weighted chance would be
1(1/ √ (1/2))(1/1.25) = 0.5667.
For FSC 13 the value would be
1(1/ √ (1/2)) (1/ √ (1/2))(1/1.25) = 0.3200, and so forth.
For tight these values would be
1(1/ √ (1/2)) (1/1.75) = 0.4041 and so on.

As the numbers in this exploration curve do not sum to one, they are not probabilities, but weights, as mentioned above. To transform the weights into probabilities, sum the values in an exploration curve, take the reciprocal, and multiply the reciprocal by each weights for the individual probabilites of finding a reservoir in a specific field size class. To calculate the overall probability of finding a reservoir of a certain FSC multiply each of the individual probabilities by the exploration success rate , in the second column, to incorporate the chance of drilling a dry hole in exploration. As an example, the probability of successfully finding and drilling a FSC 13 reservoir when the largest conventional reservoir available is size 15 is:

(Reciprocal of sum of weights) (Individual FSC weight) (Success rate)
(1 / 2.3422) (0.3200) (.14) = 0.01913 = 1.9% probability
Advanced technology will generally have a higher success rate than current.

This file can be used in modeling the effects of better seismic technology, improving the resolution of smaller field size classes. This could be done by changing (increasing) the relative probability weights of smaller(less than 10) field size classes. The exploration success rate, could also be changed to model alternative scenarios.

Note: Open the input file 'exp_dfn.spc' and initialize the specifications. etcrsq' is the exploration success rate in percentage. 'etccde' is the relative probability weight for a field size class and technology. 'etcsrt' is the exploration success rate by field size class, technology and play. 'etcflf' is the number of fields of size f1 located with exploration step f at a given technology.

```
        open(20,file='exp_dfn.spc')
        do 715 p=1,nply
        do 702 rsty=1,mxrsty
        etcrsq(rsty,p)=0.0
702     continue
        do 710 f=1,mxnfsz
        do 705 z=1,netc
        etccde(f,z)=-1.0
        etcsrt(f,z,p)=0.0
        do 703 f1=1,mxnfsz
        etcflf(f1,f,z)=0.0
703     continue
705     continue
710     continue
715     continue
```

Note: The specifications are read until the end-of-file reached. The following variables are read: Exploration technology definition name (nname), Exploration success rate (%) (esrt), Exploration curve by field size class (from field size 17 to 5) (vflf(f)).

```
720     read(20,721,end=780) nname,esrt,(vflf(f),f=1,mxnfsz)
721     format(a20,f5.1,2x,18f7.4)
```

Note: The input record (nname) is matched to the exploration technology (etcnme(z)) and if there is no match then an error message is printed.

```
730     do 735 z=1,netc
        if(nname.eq.etcnme(z)) go to 740
735     continue
        write(*,726) nname
        call errmsg(4,416)
```

Note: The exploration vector (vflf(f), i.e., the probability weight of a field size class) is initially normalized so that the first and largest reservoir size class found has a coefficient of 1.

```
740     vscl=0.0
        vxx=0.0
        do 745 f=1,mxnfsz
        if((vflf(f).ne.0.0).and.(vxx.eq.0.0)) then
        vxx=1.0/vflf(f)
        endif
        vflf(f)=vflf(f)*vxx
        vflf(f)=amax1(vflf(f),0.0)
        vscl=vscl+vflf(f)
745     continue
```

Note: The exploration vector is rescaled so that the sum of the coefficients are 1.0 and a weight is computed.

```
        if(vscl.gt.0.0) then
        valt=0.0
        vals=1.0
        do 750 f=1,mxnfsz
        vflf(f)=vflf(f)/vscl
        valt=valt+vflf(f)*vals
        vals=vals/10000.0
750     continue
        else
        valt=-0.5
        endif
```

Note: The probability weights for exploration is stored in variables.

```
        do 775 p=p1,p2
        if(p.eq.p1) then
        do 770 f=1,mxnfsz
        if(etccde(f,z).lt.valt) then
        if((etccde(f,z).ge.0.0).and.(f.lt.mxnfsz)) then
        do 760 f2=mxnfsz,f+1,-1
        do 755 f1=1,mxnfsz
        etcflf(f1,f2,z)=etcflf(f1,f2-1,z)
755     continue
        etcsrt(f2,z,p)=etcsrt(f2-1,z,p)
        etccde(f2,z)=etccde(f2-1,z)
760     continue
        elseif(etccde(f,z).ge.0.0) then
        write(*,726) nname
        call errmsg(4,417)
        endif
        etccde(f,z)=valt
        do 765 f1=1,mxnfsz
        etcflf(f1,f,z)=vflf(f1)
765     continue
        etcsrt(f,z,p)=esrt/100.0
```

Note: If a play is located in Canada (the play name begins with either the letter 'C', 'S', 'T', or 'D', or the number sequence '9002', '9004', '9005', or '9006') then the exploration success rate is hardwired to 30%.

```
        if (plynme(p)(1:1).eq.'C'.or.plynme(p)(1:1).eq.'S'.or.
        @       plynme(p)(1:1).eq.”T”.or.plynme(p)(1:1).eq.’D’.or.
        @       plynme(p)(1:4).eq.’9002’.or.plynme(p)(1:4).eq.’9004’.or.
        @       plynme(p)(1:4).eq.’9005’.or.plynme(p)(1:4).eq.’9006’.or.
        @       plynme(p)(1:4).eq.’9244’)then
        etcsrt(f,z,p) = 0.30
        endif
        go to 775
        endif
770     continue
        write(*,726) nname
        call errmsg(4,418)
        else
        do 772 f=1,mxnfsz
        etcsrt(f,z,p)=etcsrt(f,z,p1)
        if (plynme(p)(1:1).eq.'C'.or.plynme(p)(1:1).eq.'S'.or.
        @       plynme(p)(1:1).eq.”T”.or.plynme(p)(1:1).eq.’D’.or.
        @       plynme(p)(1:4).eq.’9002’.or.plynme(p)(1:4).eq.’9004’.or.
        @       plynme(p)(1:4).eq.’9005’.or.plynme(p)(1:4).eq.’9006’.or.
        @       plynme(p)(1:4).eq.’9244’)then
        etcsrt(f,z,p) = 0.30
        endif
772     continue
        endif
775     continue
        go to 720
```

Note: The input file is closed and the missing specifications are filled in for all plays, technologies, and possible vectors.

```
780     close(20)

        do 790 z=1,netc
        do 785 f=1,mxnfsz
```

Note: For an exploration vector the contribution of reservoirs of specified size class (valt) and above (valx) are computed.

```
        valt=0.0
        valx=0.0
        do 781 f1=1,mxnfsz
        if(f1.lt.f) then
        valt=valt+etcflf(f1,f,z)
        else
        valx=valx+etcflf(f1,f,z)
        endif
781     continue
```

Note: If there are no specifications for a specified size class and higher then the vector is made to look like the last one but shifted by a size class.

```
        if(valx.le.0.0) then
        if(f.le.1) then
        call errmsg(4,721)
        else
        do 783 p=1,nply
        etcsrt(f,z,p)=etcsrt(f-1,z,p)
783     continue
        do 782 f1=1,mxnfsz
        if(f1.lt.(f-1)) then
        etcflf(f1,f,z)=0.0
        elseif(f1.eq.(f-1)) then
        etcflf(f1,f,z)=0.0
        if(etcflf(f1,f-1,z).lt.1.0) then
        vxxx=1.0/(1.0-etcflf(f1,f-1,z))
        else
        vxxx=1.0
        endif
        else
        etcflf(f1,f,z)=etcflf(f1,f-1,z)*vxxx
        endif
782     continue
        endif
        endif
```

Note: A consistency check is performed. In-case of an inconsistency an error message is printed.

```
      if(etcsrt(f,z,p).gt.0.0) then
      if((etcflf(f,f,z).le.0.0).or.(valt.gt.0.0)) then
      write(*,784) plynme(p),etcnme(z),f
784   format(' Play: ',a20,' E-Technology: ',a20,' Size step: ',i3)
      call errmsg(4,418)
      endif
      endif
785   continue
790   continue
795   continue
```

Step 14: The Exploration Patterns by Exploration Increment and Technology for individual plays are read in from 'exp_ply.spc'.

Note: This file contains specifications for exploration risk by play, in the same format as EXP_DFN.SPC. Data in this play-specific file will supersede the resource type data in EXP_DFN.SPC. If EXP_PLY.SPC is missing or is of size 0, the EXP_DFN.SPC information will be used exclusively.

This file is not required to run the E&P Module. However, if desired, the exploration risk and relative probability can be specified for particular plays.

Note: The input file 'exp_ply.spc' is opened and the specification (etcflfp(f,z,p)) is initialized.

```
      open(20,file='exp_ply.spc')
      do 3715 p=1,nply
      do 3710 f=1,mxnfsz
      do 3705 z=1,netc
      etccdep(f,z,p)=-1.0
      do 3703 f1=1,mxnfsz
      etcflfp(f1,f,z,p)=0.0
3703  continue
3705  continue
3710  continue
3715  continue
      iff=0
      p=1
      z=1
```

Note: The specifications are read in until the end-of-file reached.

The following variables are read:

Play number (4-digit play code) (pname)
Technology parameter (nname)
Exploration success rate (%) (esrt)
Exploration probability weight factors for all the available field size classes (vflf(f),f=1,mxnfsz).

```
3720    continue

        read(20,3721,end=3776) pname,nname,esrt,(vflf(f),f=1,mxnfsz)

3721    format(2a20,f5.1,2x,18f7.4)

        if (pname.eq.plynme(1).and.iff.eq.0) iff=1
        if (pname.ne.plynme(p)) then
        iff=iff+1
```

Note: The input record (pname) is matched to a play (plyname(p)) and if there is no match then an error message is printed.

```
        else
        if (nname.ne.etcnme(z))then
        iff=iff+1
        go to 3730
        endif
        go to 3740
        endif
3779    do 3725 p=1,nply
        if(pname.eq.plynme(p)) go to 3730
3725    continue
        write(*,3726) pname
3726    format(' Play: ',a20)
        call errmsg(4,415)
```

Note: The input record (nname) is matched to an exploration technology (etcnme(z)) and if there is no match then an error message is printed.

```
3730    ip(iff)=p
        do 3735 z=1,netc
        if(nname.eq.etcnme(z)) go to 3740
3735    continue

        write(*,726) nname
        call errmsg(4,416)
```

Note: The exploration vector (vflf(f), i.e., the probability weight of a field size class) is initially normalized so that the first and largest reservoir size class found has a coefficient of 1.

```
3740    vscl=0.0
        vxx=0.0
        iz(iff)=z
        do 3745 f=1,mxnfsz
        if((vflf(f).ne.0.0).and.(vxx.eq.0.0)) then
        vxx=1.0/vflf(f)
        endif
        vflf(f)=vflf(f)*vxx
        vflf(f)=amax1(vflf(f),0.0)
        vscl=vscl+vflf(f)
3745    continue
```

Note: The exploration vector is rescaled so that the sum of the coefficients are 1.0 and a weight is computed.

```
        if(vscl.gt.0.0) then
        valt=0.0
        vals=1.0
        do 3750 f=1,mxnfsz
        vflf(f)=vflf(f)/vscl
        valt=valt+vflf(f)*vals
        vals=vals/10000.0
3750    continue
        else
        valt=-0.5
        endif
```

Note: The probability weights for exploration is stored in variables.

```
        do 3770 f=1,mxnfsz
        if(etccdep(f,z,p).lt.valt) then
        if((etccdep(f,z,p).ge.0.0).and.(f.lt.mxnfsz)) then
        do 3760 f2=mxnfsz,f+1,-1
        do 3755 f1=1,mxnfsz
        etcflfp(f1,f2,z,p)=etcflfp(f1,f2-1,z,p)
3755    continue
        etcsrt(f2,z,p)=etcsrt(f2-1,z,p)
        etccdep(f2,z,p)=etccdep(f2-1,z,p)
3760    continue
        elseif(etccdep(f,z,p).ge.0.0) then
        write(*,726) pname
        call errmsg(4,417)
        endif
        etccdep(f,z,p)=valt
        do 3765 f1=1,mxnfsz
        etcflfp(f1,f,z,p)=vflf(f1)
3765    continue
        etcsrt(f,z,p)=esrt/100.0
        go to 3775
        endif
3770    continue
        write(*,726) nname
        call errmsg(3,3418)
3775    continue
        go to 3720
3776    continue
```

Note: The input file is closed and the missing specifications are filled in for all plays, technologies, and possible vectors.

```
        CLOSE(20)

        do 3795 iip=1,iff
        p=ip(iip)
        z=iz(iip)
        do 3785 f=1,mxnfsz
```

Note: For an exploration vector the contribution of reservoirs of specified size class (valt) and above (valx) are computed.

```
        valt=0.0
        valx=0.0
        do 3781 f1=1,mxnfsz
        if(f1.lt.f) then
        valt=valt+etcflfp(f1,f,z,p)
        else
        valx=valx+etcflfp(f1,f,z,p)
        endif
3781    continue
```

Note: If there are no specifications for a specified size class and above then the vector is made to look like the last one but shifted by a size class.

```
        if(valx.le.0.0) then
        if(f.le.1) then
        call errmsg(4,721)
        else
        do 3782 f1=1,mxnfsz
        if(f1.lt.(f-1)) then
        etcflfp(f1,f,z,p)=0.0
        elseif(f1.eq.(f-1)) then
        etcflfp(f1,f,z,p)=0.0
        if(etcflfp(f1,f-1,z,p).lt.1.0) then
        vxxx=1.0/(1.0-etcflfp(f1,f-1,z,p))
        else
        vxxx=1.0
        endif
        else
        etcflfp(f1,f,z,p)=etcflfp(f1,f-1,z,p)*vxxx
        endif
3782    continue
        endif
        endif
```

Note: A consistency check is performed. In-case of an inconsistency anerror message is printed.

```
        if(etcsrt(f,z,p).gt.0.0) then
        if((etcflfp(f,f,z,p).le.0.0).or.(valt.gt.0.0)) then
        write(*,784) plynme(p),etcnme(z),f
        call errmsg(4,3419)
        endif
        endif
3785    continue
3790    continue
3795    continue
```

Step 15: The Resource Availabiliy Curves for Undiscovered Resources are read in from 'resav.spc'.

Note: The variable representing the reserve availability factor (vscl) is first initialized and then 'resav.spc' is opened and the name of resource types is read (nmrsty).

```
        vscl=0

        open(18,file='resav.spc')
        do itype=1,mxrsty
        read(18,2333) nmrsty(itype)
2333    format(a9)
        enddo
```

Note: This code initializes the availability penetration parameter to –1 for all years, resource types and supply regions.

```
        Do ijk=1,mxnyr
        Do ijkres=1,mxrsty
        Do ijksup=1,mxnsrg
        availpen(ijk,ijkres,ijksup)= -1.0
        Enddo
        Enddo
        Enddo
```

Note: The 'resav.spc' file is read, i.e., the resource type (resname), the year and the availability factor (vscl). A pointer (y) is set up and the reserve growth factor (vscl) is saved into ‘availpen(y,ijkres,ijksup)’.

```
        read(18,*,end=289)

299     read(18,301,end=350) resname,year,vscl
        if(year.eq.-1) goto 300
        do ichrsty=1,mxrsty
        if(resname.eq.nmrsty(ichrsty)) then
        ijkres=ichrsty
        go to 298
        endif
        enddo
298     y=year-tmex+1
        Do ijksup=1,mxnsrg
        availpen(y,ijkres,ijksup)=vscl/100.0
        Enddo
        go to 299
289     print *,'Availability file resav.spc invalid'
        stop
```

Note: Here it is confirmed that the region name in 'resav.spc' is valid.

```
300     continue
        read(18,'(a20)',end=350) regname
        call clook20(regname,srgnme,mxnsrg,iclook)
        if (iclook.eq.0) then
        print*, 'Problem in Look-Up of Resource Availability File'
        write(*,*) regname
        stop
        endif
```

Note: The relevant variables (i.e., resource name, year and availability factor for a region) are read for all the years.

```
302     read(18,301,end=350) resname,year,vscl
301     format(a9,t10,i4,f6.1)
        if(year.eq.-1) goto 300
        do ichrsty=1,mxrsty
        if(resname.eq.nmrsty(ichrsty))then
        ijkres=ichrsty
        go to 297
        endif
        enddo
297     y=year-tmex+1
```

Note: The reserve availability factor is saved and the next record is analyzed.

```
        availpen(y,ijkres,iclook)=vscl/100.0
        go to 302

350     close(18)

        do ijksup=1,mxnsrg
        do ijkres=1,mxrsty
        if(availpen(mxnyr,ijkres,ijksup).ge.0.0)go to 303
        do ichkyr=mxnyr-1,1,-1
        if(availpen(ichkyr,ijkres,ijksup).ge.0.0)then
        availpen(mxnyr,ijkres,ijksup)=
     &  availpen(ichkyr,ijkres,ijksup)
        goto 303
        endif
        enddo
303     continue
        enddo
        enddo
```

Note: Data is filled in for missing years using interpolation.

```
        Do 390 isup=1,mxnsrg

        do 390 x=1,mxrsty
        y1=0
        y2=0

        do 380 y=1,mxnyr
        if(availpen(y,x,isup).ne.-1.0) then

        y1=y
        else

        if(y2.lt.y) then
        do 360 y2=y,mxnyr
        if(availpen(y2,x,isup).ne.-1.0) go to 365
360     continue
        y2=mxnyr+1
365     continue
        endif
        if (y2.gt.mxnyr) y2=mxnyr
```

Note: For the missing years the availability factor between the years for which data is specified is interpolated (linearly) to get the value for the missing year. It is assumed that the availability factor remains constant after the last year specified. If the data is not yet specified then the availability factor is set to zero.

```
        if((y1.eq.0).and.(y2.gt.mxnyr)) then
        write(*,366)
```

```
         print *, y1,y2
366      format( '  Availability Problems',a20)
         call errmsg(4,422)
         elseif(y1.eq.0) then
         availpen(y,x,isup)=availpen(y2,x,isup)
         elseif(y2.gt.mxnyr) then
         availpen(y,x,isup)=availpen(y1,x,isup)
         else
         vscl=float(y2-y)/float(y2-y1)
         availpen(y,x,isup)=vscl*availpen(y1,x,isup) +
    &    (1.0-vscl)* availpen(y2,x,isup)
         endif
         endif
380      continue
390 c    ontinue
```

Step 16: **The Availability Curves for Reserves Growth are read in from 'resavrg.spc'.**

Note: The variable representing the reserve availability factor (vscl) is first initialized and then 'resavrg.spc' is opened and the name of resource types is read (nmrstyrg).

```
         vscl=0

         open(18,file='resavrg.spc')
         do itype=1,mxrsty
         read(18,2333) nmrstyrg(itype)
         if(nmrstyrg(itype).ne.nmrsty(itype))print *,'Resource Mismatch'

         enddo
```

Note: This code initializes the availability penetration parameter to –1 for all years, resource types and supply regions.

```
         Do ijk=1,mxnyr
         Do ijkres=1,mxrsty
         Do ijksup=1,mxnsrg
         availpen_rg(ijk,ijkres,ijksup)= -1.0
         Enddo
         Enddo
         Enddo
```

Note: The 'resavrg.spc' file is read, i.e., the resource type (resname), the year and the reserve availability factor (vscl). A pointer (y) is set up and the reserve growth factor (vscl) is saved into ‘availpen_rg(y,ijkres,ijksup)’.

```
        read(18,*,end=3289)

3299    read(18,301,end=3350) resname,year,vscl
        if(year.eq.-1) goto 3300
        do ichrsty=1,mxrsty
        if(resname.eq.nmrsty(ichrsty))then
        ijkres=ichrsty
        go to 3298
        endif
        enddo
3298    y=year-tmex+1
        Do ijksup=1,mxnsrg
        availpen_rg(y,ijkres,ijksup)=vscl/100.0
        Enddo
        go to 3299
3289    print *,'Availability file res_av.spc invalid'
        stop
```

Note: Here it is confirmed that the region name in 'resavrg.spc' is valid.

```
3300    read(18,'(a20)',end=3350) regname
        call clook20(regname,srgnme,mxnsrg,iclook)
        if (iclook.eq.0) then
        print*, 'Problem in Look-Up of Resource Availability File'
        write(*,*) regname
        stop
        endif
```

Note: The relevant variables (i.e., resource name, year and availability factor for a region) are read for all the years.

```
3302    read(18,301,end=3350) resname,year,vscl
        if(year.eq.-1) goto 3300
        do ichrsty=1,mxrsty
        if(resname.eq.nmrsty(ichrsty))then
        ijkres=ichrsty
        go to 3297
        endif
        enddo
3297    y=year-tmex+1
```

Note: The reserve availability factor is saved and the next record is analyzed.

```
        availpen_rg(y,ijkres,iclook)=vscl/100.0
        go to 3302
3350    close(18)


        do ijksup=1,mxnsrg
        do ijkres=1,mxrsty
        if(availpen_rg(mxnyr,ijkres,ijksup).ge.0.0)go to 3303
        do ichkyr=mxnyr-1,1,-1
        if(availpen_rg(ichkyr,ijkres,ijksup).ge.0.0)then
        availpen_rg(mxnyr,ijkres,ijksup)=
     &  availpen_rg(ichkyr,ijkres,ijksup)
        goto 3303
        endif
        enddo
3303    continue
        enddo
        enddo
```

Note: Data is filled in for missing years using interpolation.

```
        Do 3390 isup=1,mxnsrg

        do 3390 x=1,mxrsty
        y1=0
        y2=0

        do 3380 y=1,mxnyr
        if(availpen_rg(y,x,isup).ne.-1.0) then

        y1=y
        else

        if(y2.lt.y) then
        do 3360 y2=y,mxnyr
        if(availpen_rg(y2,x,isup).ne.-1.0) go to 3365
3360    continue
        y2=mxnyr+1
3365    continue
        endif
        if (y2.gt.mxnyr) y2=mxnyr
```

Note: For the missing years the availability factor between the years for which data is specified is interpolated (linearly) to get the value for the missing year. It is assumed that the availability factor remains constant after the last year specified. If the data is not yet specified then the availability factor is set to zero.

```
        if((y1.eq.0).and.(y2.gt.mxnyr)) then
        write(*,366)
        print *, y1,y2
        call errmsg(4,422)
        elseif(y1.eq.0) then
        availpen_rg(y,x,isup)=availpen_rg(y2,x,isup)
        elseif(y2.gt.mxnyr) then
        availpen_rg(y,x,isup)=availpen_rg(y1,x,isup)
        else
        vscl=float(y2-y)/float(y2-y1)
        availpen_rg(y,x,isup)=vscl*availpen_rg(y1,x,isup) +
     &      (1.0-vscl)* availpen_rg(y2,x,isup)
            endif
            endif
3380        continue
3390        continue
```

Step 17: **The development technology penetration specifications are read in from 'dtec_pen.spc'.**

Note: This file shows the market penetration rate for development technologythrough time for current and advanced technology. Generally current technology penetrates more and earlier than advanced technology.

A value of 100 for the non-drilling cost factor means that operating and other non-drilling costs are the same as specified in the Reservoir Performance Module. The cost factor can be used to reduce (or increase) the non-drilling cost over time.

Development technology affects both the decision to explore (because the economics of exploration are directly related to the cost and recovery efficiencies of ultimate development practices) and the rate of development of a given resource. Hence the exploitation of undiscovered resource depends both on exploration and development technology penetration rates and the amount developed is constrained by the development technology penetration rates. For the discovered producing category development technology penetration curves affect the infill drilling and completion opportunities.

Development technology penetration rates are used in GSAM to reflect operator acceptance over time of emerging technologies. Technology impacts can be delayed (by setting a year's penetration rate to 0 or a very low value) to reflect the time required to develop, test, and implement new practices. Further, technology penetration for a given technology and resource can be flat or declining to reflect market saturation of a technology or to force a switch to an emerging technology from a less efficient method. The non-drilling costs can be varied to model the higher costs associated with initial applications of a given practice and the trend of falling costs with time as the technology is more widely understood and applied.

The application of any technology modeled in the Reservoir Performance Module can be varied using development technology penetration rates. This includes changes in skin factors, alternative hydraulic fracturing methods, lower (or higher) drilling costs, changes in completion methods, and alternative operating practices and costs.

DTEC_PEN.SPC can be used to measure the impact of increasing or decreasing a development technology's market penetration at any point in time. The cost factor parameter can be used to study the decrease/increase of non-drilling cost as a function of time. It should be realized that these penetration curves are applied to all features of development technology modeled in the Reservoir Performance Module.

Note: The number of development technologies (ndtc) is initialized along with the variable representing the penetration rates (vscl) and the input file 'dtec_pen.spc' is opened. The input specifications are read in until the end-of-file is reached. The input variables include:

- Development technology parameter name (nname)
- Time period (year)
- Technology penetration rate (%) (vscl)
- Non-drilling cost factor (vcst)

```
        open(21,file='dtec_pen.spc')

        ndtc=0
        vscl=0

800     read(21,801,end=850) nname,year,vscl,vcst
801     format(a20,i4,2f6.1)
```

Note: The development technology rate (vscl) is forced into valid ranges (i.e., between zero and hundred) and if the cost scaling (vsct) not specified, it is made equal to 100.0

```
        vscl=amax1(0.0,amin1(vscl,100.0))
        if(vcst.eq.0.0) vcst=100.0
```

Note: The development technology is located and the penetration parameters (dtcpen) for it are initialized.

```
        x=0
810     x=x+1
        if(x.gt.ndtc) go to 820
        if(nname.ne.dtcnme(x)) go to 810
        go to 830
820     ndtc=ndtc+1
```

```
        if(ndtc.gt.mxndtc) call errmsg(4,419)
        dtcnme(x)=nname
        do 825 y=1,mxnyr
        dtcpen(y,x)=-1.0
825     continue
```

Note: It is ensured that the year is valid and that the penetration has not already been specified for that year.

```
830     y=year-tmex+1
        if((y.le.0).or.(y.gt.mxnyr)) then
        write(*,831) nname,year
        call errmsg(4,420)
        endif
        if(dtcpen(y,x).ne.-1.0) then
        write(*,831) nname,year
831     format(' Dev-Technology: ',a20,' Year: ',i4)
        call errmsg(4,421)
        endif
```

Note: The penetration and cost scaling is saved (vscl as dtcpen and vcst as dtccsf) and the next record is processed.

```
        dtcpen(y,x)=vscl/100.0
        dtccsf(y,x)=vcst/100.0
        go to 800
```

Note: The input file is closed and it is confirmed that at least one development technology is specified.

```
850     close(21)

        if(ndtc.le.0) call errmsg(4,422)
```

Note: Data is filled in for missing years using interpolation.

```
        do 890 x=1,ndtc
        y1=0
        y2=0

        do 880 y=1,mxnyr
        if(dtcpen(y,x).ne.-1.0) then

        y1=y
        else
        if(y2.lt.y) then
        do 860 y2=y,mxnyr
        if(dtcpen(y2,x).ne.-1.0) go to 865
860     continue
        y2=mxnyr+1
865     continue
        endif
```

Note: For the missing years the availability factor between the years for which data is specified is interpolated (linearly) to get the value for the missing year. It is assumed that the availability factor remains constant after the last year specified. If the data is not yet specified then the availability factor is set to zero.

```
        if((y1.eq.0).and.(y2.gt.mxnyr)) then
        write(*,866) dtcnme(x)
866     format(' Dev-Technology: ',a20)
        call errmsg(4,422)
        elseif(y1.eq.0) then
        dtcpen(y,x)=dtcpen(y2,x)
        dtccsf(y,x)=dtccsf(y2,x)
        elseif(y2.gt.mxnyr) then
        dtcpen(y,x)=dtcpen(y1,x)
        dtccsf(y,x)=dtccsf(y1,x)
        else
        vscl=float(y2-y)/float(y2-y1)
        dtcpen(y,x)=vscl*dtcpen(y1,x)+(1.0-vscl)*dtcpen(y2,x)
        dtccsf(y,x)=vscl*dtccsf(y1,x)+(1.0-vscl)*dtccsf(y2,x)
        endif
        endif
880     continue
890     continue
```

Step 18: **The Federal Lands Technology Penetration Increments are read in for development technology (from 'dtec_fed.spc') and exploration technology (from 'etec_fed.spc').**

Note: The 'dtec_fed.spc' and 'etec_fed.spc' files are opened and the incremental Federal Lands Penetration values are read for Development Technology and Exploration Technology respectively. The following variables are read in: Year, Current Technology Increment (vscl_c), Advanced Technology Increment (vscl_a). There are three header lines in the files. The federal lands technology penetration increments should be specified for all years. No interpolation routine exists to perform interpolation as was done in 'dtec_pen.spc/etec_pen.spc.'

```
        open(17,file='dtec_fed.spc')
        open(18,file='etec_fed.spc')
        read(17,*)
        read(17,*)
        read(17,*)
        read(18,*)
        read(18,*)
        read(18,*)

648     read(17,*,end=649) year,vscl_c,vscl_a
```

Note: It is ensured that the year is not out of bounds.

```
        y=year-tmex+1
        if (year.gt.tme(ntme)) goto 648
        if (y.le.0.or.y.gt.mxnyr) then
        print *, ' Problem in dtec_fed.spc File: Year Out of Bounds '
        stop
        endif
```

Note: The incremental Federal Lands Penetration values are saved in the variable 'fedpend'. The code checks to ensure that total technology penetration (i.e., development technology penetration plus federal lands increments) is within zero and a hundred percent. The file is then closed.

```
        fedpend(y,1) = vscl_c/100
        fedpend(y,2) = vscl_c/100
        fedpend(y,3) = vscl_c/100
        fedpend(y,4) = vscl_c/100
        fedpend(y,5) = vscl_c/100
        fedpend(y,6) = vscl_c/100
        fedpend(y,7) = vscl_c/100
        fedpend(y,8) = vscl_a/100
        fedpend(y,9) = vscl_a/100
        fedpend(y,10) = vscl_a/100
        fedpend(y,11) = vscl_a/100
        fedpend(y,12) = vscl_a/100
        fedpend(y,13) = vscl_a/100
        fedpend(y,14) = vscl_a/100

        if ((fedpend(y,1) + dtcpen(y,1)).lt.0.0) then
        print *, ' Federal Increment is Low:: -Check dtec_fed.spc File'
        print *, ' Check Current Development Technology Numbers'
        print *, ' Problem in Year =', y+tmex-1
        stop
        endif

        if ((fedpend(y,2) + dtcpen(y,2)).lt.0.0) then
        print *, ' Federal Increment is Low::- Check dtec_fed.spc File'
        print *, ' Check Advanced Development Technology Numbers'
        print *, ' Problem in Year =', y+tmex-1
        stop
        endif
        if ((fedpend(y,1) + dtcpen(y,1)).gt.1.0) then
        print *, ' Federal Increment is High::- Check dtec_fed.spc File'
        print *, ' Check Current Development Technology Numbers'
        print *, ' Problem in Year =', y+tmex-1
        stop
        endif

        if ((fedpend(y,2) + dtcpen(y,2)).gt.1.0) then
        print *, ' Federal Increment is High::- Check dtec_fed.spc File'
        print *, ' Check Advanced Development Technology Numbers'
        print *, ' Problem in Year =', y+tmex-1
        stop
        endif

        goto 648
649     close(17)
```

Note: The above steps are repeated for the exploration technology.

```
698     read(18,*,end=699) year,vscl_c,vscl_a

        y=year-tmex+1
        if (year.gt.tme(ntme)) goto 698

        if (y.le.0.or.y.gt.mxnyr) then
        print *, ' Problem in etec_fed.spc File: Year Out of Bounds '
        stop
        endif

        fedpene(y,1) = vscl_c/100
        fedpene(y,2) = vscl_c/100
        fedpene(y,3) = vscl_c/100
        fedpene(y,4) = vscl_c/100
        fedpene(y,5) = vscl_c/100
        fedpene(y,6) = vscl_c/100
        fedpene(y,7) = vscl_c/100
        fedpene(y,8) = vscl_a/100
        fedpene(y,9) = vscl_a/100
        fedpene(y,10) = vscl_a/100
        fedpene(y,11) = vscl_a/100
        fedpene(y,12) = vscl_a/100
        fedpene(y,13) = vscl_a/100
        fedpene(y,14) = vscl_a/100

        if ((fedpene(y,1) + etcpen(y,1)).lt.0.0) then
        print *, ' Federal Increment is Low::- Check etec_fed.spc File'
        print *, ' Check Current Exploration Technology Numbers'
        print *, ' Problem in Year =', y+tmex-1
        stop
        endif

        if ((fedpene(y,2) + etcpen(y,2)).lt.0.0) then
        print *, ' Federal Increment is Low::- Check etec_fed.spc File'
        print *, ' Check Advanced Exploration Technology Numbers'
        print *, ' Problem in Year =', y+tmex-1
        stop
        endif
        if ((fedpene(y,1) + etcpen(y,1)).gt.1.0) then
        print *, ' Federal Increment is High::- Check etec_fed.spc File'
        print *, ' Check Current Exploration Technology Numbers'
        print *, ' Problem in Year =', y+tmex-1
        stop
        endif

        if ((fedpene(y,2) + etcpen(y,2)).gt.1.0) then
        print *, ' Federal Increment is High::- Check etec_fed.spc File'
        print *, ' Check Advanced Exploration Technology Numbers'
        print *, ' Problem in Year =', y+tmex-1
        stop
        endif

        goto 698
699     close(18)
```

Step 19: **The file is used to specify the year in which royalty incentive would start to be implemented.**

Note: The input specifications are read:

year
Tax regime (currently modeled as royalty incentive start year) ('h')

The variable of significance is taxcde(y). The ‘year’ specified in tax_cde.spc file is assigned the ‘taxcde(y)’ variable.

```
1020    read(23,1021,end=1050) year,h
1021    format(i4,i2)

        do  y=1,mxnyr
        if ( y.lt.(year-tmex+1) ) then
        taxcde(y) = 0
        else
        taxcde(y) = 1
        endif
        enddo
```

Step 20: **Development Technology Parameters are read in from 'dvl_tpr.spc' and mapped to resource types.**

Note: The input file is opened and the specifications are initialized (nrst representing the resource type is set to zero and xdeqv representing the level of technology is set to -1).

```
        open(20,file='dvl_tpr.spc')
        nrst=0
        do 1210 i=1,mxrsty
        do 1205 x=1,mxndtc
        xdeqv(x,i)=-1
1205    continue
1210    continue
```

Note: The input specifications are read in until the end-of-file reached. The variables include:

The development technology parameter name (nname)
Resource type (values range from 1 to 7), ('i')
Flag for level of technology (current = 0; advanced = 1), (x1)

```
1220    read(20,1221,end=1240) nname,i,x1
1221    format(a20,i4,i6)
```

Note: The input (nname) is matched to a development technology (dtcnme) and an error is printed if no match is found.

```
        do 1230 x=1,ndtc
        if(nname.eq.dtcnme(x)) go to 1235
1230    continue
        call errmsg(4,431)
```

Note: The maximum number of resource types (nrst) is saved and its validity is ensured.

```
1235    nrst=max0(nrst,i)
        if(nrst.gt.mxrsty) call errmsg(4,432)
```

Note: First it is made sure that the technology mapping is not a duplicate (incase it is a duplicate an error message is printed) and then the mapping is saved (x1 as xdeqv(x,i)).

```
        if(xdeqv(x,i).ne.-1) then
        call errmsg(4,433)
        endif
        xdeqv(x,i)=x1
        go to 1220
```

Note: The input file is closed and the array that points from technology option (1-current, 2-advanced) for each resource type to actual technology is initialized (xdeqx is set to zero).

The pointers from primary/secondary combinations of technology options to actual technologies are also initialized (xdeqy is set to zero).

```
1240    close(20)
        do 1250 i=1,nrst
        do 1241 wi=1,mxndtx
        xdeqx(wi,i)=0
1241    continue
        wj=0
        do 1243 w1=1,mxndtx
        do 1242 w2=w1,mxndtx
        wj=wj+1
        xdeqy(1,wj,i)=0
        xdeqy(2,wj,i)=0
1242    continue
1243    continue
```

Note: It is ensured that at least one technology is specified for the resource type and that the maximum number is not exceeded.

```
        x1=0
        do 1245 x=1,ndtc
        if(xdeqv(x,i).ge.0) then
        x1=x1+1
        endif
1245    continue
        if((x1.le.0).or.(x1.gt.mxndtx)) then
        write(*,1246) i,x1,wi,(xdeqv(x,i),x=1,ndtc)
1246    format(' reservoir type: ',3i3/10(1x,i3))
        call errmsg(4,912)
        endif
```

Note: The technology for the option is located by looping through all the different technology options, the pointer is saved from the option to the technology (in the ‘xdeqx’ variable). If the technology for the option is not found an error message is printed.

```
        do 1248 wi=1,x1
        do 1247 x=1,ndtc
        if(xdeqv(x,i).ne.(wi-1)) go to 1247
        xdeqx(wi,i)=x
        go to 1248
1247    continue
        write(*,1246) i,x1,wi,(xdeqv(x,i),x=1,ndtc)
        call errmsg(4,912)
1248    continue
```

Note: The pointers (xdeqy(1,wj,i); xdeqy(2,wj,i)) are saved for the secondary technology and are set to be the same as that of the primary technology.

```
      wj=0
      do 1249 wi=1,x1
      do 1244 w2=wi,x1
      wj=wj+1
      xdeqy(1,wj,i)=xdeqx(wi,i)
      xdeqy(2,wj,i)=xdeqx(w2,i)
1244  continue
1249  continue
1250  continue
```

Step 21: This subroutine compares two variables from different sources.

```
      subroutine clook20(code,array,n,i)
      integer*4 n,i
      character*20 code,array(*)

      do i=1,n
      if(code.eq.array(i)) return
      enddo
      i=0
      return
      end
```

Table of Contents

SUBROUTINE EXDVI2

CALLED BY:	EXPLPROD (Controls the main flow through the expl./dvlp. model.)
CALLS:	GETT (Estimates the time required for each phase of the subroutine.) ERRMSG (Prints out errors and warnings)
READS:	**‘**undbnk.spc’ (Contains the developed fraction of undiscovered resource.) ‘disb.bnk’ (Contains development profiles for known fields.) ‘dis.bnk’ (Contains development profiles for known fields.) ‘undb.bnk’ (Contains development profiles for undiscovered resources.)

MAIN THEME: This routine reads in various parameters which are stored in the binary files. The routine performs internal consistency checks and in the event when the checks fail prints error messages on the screen. The data also gets re-processed and converted into useable variables before being passed over to other routines of the E&P module.

The general format followed while reading the three main files in this subroutine follows the following sequence:
a) The data is read.
b) Internal consistency checks are performed to make sure that the data is consistent, if it isn't error messages are printed.
c) The data is re-processed and stored in an acceptable format.

Step 1: **The file 'undbnk.spc' is opened.**

Note: This file has been developed to take into account the developed fraction of undiscovered resource. It essentially contains two factors: one which
accounts for undeveloped resource by region and FSC and another factor which gets multiplied to the undiscovered resource estimates. The values (in the 2nd column) could be used to increase/decrease the undiscovered resource estimates to be within different confidence intervals (such as 25%, 75% etc.) of the USGS
resource estimates. The factors to the right are availability by field size class. As an example, the MAFLA onshore

region (counter #2) has 1.10, 0.18, and 0.18 in the second, third, and fourth columns. The factor on the left side (in this case 1.1) multiplies the total undiscovered resource by this amount, usually to "make up for" the resource that is now considered undeveloped. The second and third factors of 18% mean that 18% of the undiscovered resource in FSC 17 and 18% of the undiscovered resource in FSC 16 are actually undeveloped.

The two numbers at the bottom of the file represent the Canadian to U.S. dollar exchange rates. The numbers indicate that the Canadian dollar is 0.75/0.85 = 88% in value compared to the U.S. dollar.

UNDBNK.SPC is used to adjust resource estimates post- Resource and RP Module, and is only place that the U.S. discovered undeveloped is accounted for. It can be used to test the sensitivity of future market estimates to an increased measure of the resource base, by FSC and region, or by undeveloped and undiscovered resource.

```
open(33,file='undbnk.spc')
```

Step 2: **The variables representing the exchange rate between Canada and the U.S. are initialized and the 'undbnk.spc' is read.**

Note: The following variables are read:

1) GSAM Supply Region Counter (igrsr)
2) Undiscovered resource factor (a factor of 1.1 means the undiscovered resource is 10% more than the Resource Module estimation), (resfac(igrsr)).
3) Percent of undiscovered resource which is discovered but undeveloped, by Field Size Class (17 to 5), (curbnk(ifsxze,igrsr),ifsxze=1,mxnfsz)
4) The exchange rate variables (exchrate,exchbase)

The exchange rate is then calculated (exchrate).

```
exchbase=1.0
exchrate=1.0

read(33,*)
do igr=1,mxnsrg
read(33,*,end=222)igrsr,resfac(igrsr),(curbnk(ifsxze,igrsr)
        $,ifsxze=1,mxnfsz)
        enddo

        read(33,*) exchrate,exchbase
222     close(33)
        exchrate = exchrate/exchbase
```

Step 3: **The undiscovered data bank (binary file) 'news\undb.bnk' is opened.**

Note: The two header lines are read and stored in the dummy variable 'ctch'. The following variables are initialized:

1) Pointers (such as 'po','pnameo','v','ctsto','ii').
2) Number of economic new field play-field size categories (nnfl)
3) Pointer to economic new field play/field-size category (nnflp(f,p))
4) Original no. of reservoirs to be discovered in FSC (nnflpo(f,p))
5) Remaining no. of reservoirs to be discovered in FSC (nnflpr(f,p))
6) The back pointer to the previous undiscovered reservoir (nnflb(g))
7) Resource type for undiscovered reservoirs (ndrsty(g))
8) Total footage needed to be drilled for all undiscovered reservoirs in FSC of the play (nddpth(g))
9) For an undiscovered reservoir the year when production starts to decline (ndwin(wi,g))
10) Number of wells in an undiscovered reservoir (ndnwl(d,g))
11) Present value of production for all the undiscovered reservoirs in FSC of the play (ndpvp(b,d,wi,g))
12) Npv of non drilling costs at $2.00 gas price for all reservoirs in FSC of the play it includes NPV of non-drilling costs as calculated in Reservoir Performance Module + NPV of expenses (ndpvnd(b,d,wi,g))
13) Npv of drilling costs at $2.00 gas price and base full drilling costs (ndpvdc(b,d,wi,g))
14) Npv of taxes at $2.00 gas price and base full drilling costs (ndpvtx(b,d,wi,g))
15) Increase in NPV of non-drilling cost for increase in gas price of $1.00 (ndpcnd(b,d,wi,g))
16) Increase in NPV of drilling cost for increase in gas price of $1.00 (ndpcdc(b,d,wi,g))
17) Increase in NPV of taxes for increase in gas price of $1.00 (ndpctx(b,d,wi,g))
18) Drilling cost slope (ndpvds(b,d,wi,g))
19) Non-drilling cost slope (ndpvns(b,d,wi,g))

20) Number of years a reservoir can technically produce (ndpryr(b,d,wi,g))
21) Original Gas In Place (ndogip(d,g))
22) A pointer for new and undiscovered reservoirs to the next one in the same FSC (kpnt(v))
23) Reservoir type for discovered reservoirs (kdrsty(v))
24) Reservoir depth for discovered reservoirs (kddpth(v))
25) For a discovered reservoir year that window opens (relative to the starting year of production) (kdwin(wi,v))
26) Number of wells in a discovered reservoir (kdnwl(1,d,v))
27) Present value of production for all the discovered reservoirs in FSC of the play (kdpvp(b,d,wi,v))

```
        po=0
        pnameo='                    '
        1100  open(24,file='news\undb.bnk',form='BINARY')

        do 1110 p=1,nply
        do 1105 f=1,mxnfsz
        nnflp(f,p)=0
        nnflpo(f,p)=0.0
        nnflpr(f,p)=0.0
1105    continue
1110    continue

        do 1120 g=1,mxnnfl
        nnflb(g)=0
        ndrsty(g)=0
        nddpth(g)=0
        do 1116 wi=1,mxndty
        ndwin(wi,g)=0
        do 1114 d=1,mxngr
        ndnwl(d,g)=0
        do 1112 b=1,mxdevt
        ndpvp(b,d,wi,g)=-1.0
        ndpvnd(b,d,wi,g)=0.0
        ndpvdc(b,d,wi,g)=0.0
        ndpvtx(b,d,wi,g)=0.0
        ndpcnd(b,d,wi,g)=0.0
        ndpcdc(b,d,wi,g)=0.0
        ndpctx(b,d,wi,g)=0.0
        ndpvds(b,d,wi,g)=0.0
        ndpvns(b,d,wi,g)=0.0
        ndpryr(b,d,wi,g)=0
        if((b.eq.1).and.(wi.eq.1)) then
        ndogip(d,g)=0.0
        endif
1112    continue
1114    continue
1116    continue
1120    continue
        v=0
        nnfl=0
        ctsto='    '
        ii=0
        1130 ii=ii+1
        if(ii.le.2) then
        v=v+1
        if(iocde.eq.0) read(24,1129,end=1210) ctch
        1129  format(a1)
        if(iocde.eq.0) read(24,1129,end=1210) ctch
        ii=ii+2
        if(v.gt.mxnefl) call errmsg(4,451)
        kpnt(v)=0
        kdrsty(v)=0
        kddpth(v)=0
        do 1136 wi=1,mxndty
        kdwin(wi,v)=0
        do 1134 d=1,mxngr
        kdnwl(1,d,v)=0
        do 1132 b=1,mxdevt
        kdpvp(b,d,wi,v)=-1.0
1132    continue
1134    continue
1136    continue
        elseif(ii.eq.20) then
        ii=0
        endif
```

Step 4: **The Development Profiles are read from the 'news\undb.bnk' file by Play, Field Size, and Technology.**

Note: The following variables are read:

1) Supply region code (ipn)
2) Status of resource (c1)
3) Resource Type (rsty)
4) USGS Play number (ctst)
5) Field size class of a GSAM ID for an undiscovered reservoir (f1)
6) Technology case (ctch)
7) Paygrade (d)
8) Development type case (Primary, Refrac., Infill), (copt)
9) Technically recoverable reserves (Bcf), (xres)
10) Original gas in place (xogip)
11) Number of wells (xnw)
12) MASP (xmasp)
13) Total capacity (xtcap)
14) NPV of production in Bcf (xpvp)
15) NPV of expenses in million $ (xpvec)
16) NPV of investment in million $ (xpvtc)
17) NPV of drilling cost in million $ (xpvdc)
18) NPV of non-drilling cost in million $ (xpvndc)
19) Severance, federal and state taxes in million $ (xpvtax)
20) Changes in expenses as the price of gas goes from $5 to $2 (ratio), (xpcec)
21) Changes in investment as the price of gas goes from $5 to $2

(ratio), (xpctc)
22) Changes in taxes as the price of gas goes from $5 to $2 (ratio), (xpctax)
23) Drill slope (xpvds)
24) Non-drilling slope (xpvnds)
25) Reservoir depth (idpth)
26) Water depth (ih2o)
27) Window year (iwin)
28) Productive life (ipryr)
29) Number of undiscovered reservoirs in the field size class of the play (nrrr)

```
1137    read(24,end=1210)   ipn,c1,rsty,ctst,f1,ctch,d,copt,xres,xogip,
        xnw,xmasp,xtcap,xpvp,xpvec,xpvtc,xpvdc,xpvndc,xpvtax,
        xpcec,xpctc,xpctax,xpvds,xpvnds,idpth,ih2o,iwin,
        ipryr,nrrr
```

Step 5: **Here it is ensured that there is no inconsistency between the supply region code read from the 'news\undb.bnk' (ipn) and that from the 'undbnk.spc' (igrsr).**

```
        igrsr=6
        if(ipn.eq.'24')igrsr=24
        if(ipn.eq.'23')igrsr=22
        if(ipn.eq.'22')igrsr=21
        if(ipn.eq.'19')igrsr=19
        if(ipn.eq.'18')igrsr=18
        if(ipn.eq.'17')igrsr=17
        if(ipn.eq.'16')igrsr=16
        if(ipn.eq.'15')igrsr=15
        if(ipn.eq.'14')igrsr=14
        if(ipn.eq.'13')igrsr=13
        if(ipn.eq.'12')igrsr=12
        if(ipn.eq.'11')igrsr=11
        if(ipn.eq.'10')igrsr=10
        if(ipn.eq.'09')igrsr=9
        if(ipn.eq.'08')igrsr=8
        if(ipn.eq.'07')igrsr=7
        if(ipn.eq.'06')igrsr=6
        if(ipn.eq.'05')igrsr=5
        if(ipn.eq.'04')igrsr=4
        if(ipn.eq.'03')igrsr=3
        if(ipn.eq.'02')igrsr=2
        if(ipn.eq.'01')igrsr=1
```

Step 6: **Here there is a provision for the number of undiscovered reservoirs (nrrr) to be altered from their previous discovered value by using the factor 'resfac' from 'undbnk.spc'**

```
nrrr=nint(nrrr*resfac(igrsr)+.49)
```

Step 7: **Here it is ensured that if the value of the window year calculated in the Reservoir Performance module is less than 2 the variable representing the window year (iwin) is hardwired to equal two years.**

```
if(iwin.lt.2) iwin=2
```

Step 8: **The following variables are recalibrated so that the production variables are in MMSCF and cost variables are in thousand dollars.**

```
xres=xres*1000.0
xogip=xogip*1000.0
xtcap=xtcap*1000.0
xpvp=xpvp*1000.0
xpvec=xpvec*1000.0
xpvtc=xpvtc*1000.0
xpvdc=xpvdc*1000.0
xpvndc=xpvndc*1000.0
xpvtax=xpvtax*1000.0
xpcec=xpcec*1000.0
xpctc=xpctc*1000.0
xpctax=xpctax*1000.0
```

Step 9: **The expense variables for Canadian reservoirs are adjusted using the exchange rate.**

```
if(ipn.eq.'22'.or.ipn.eq.'23'.or.ipn.eq.'24')then
xpvec=xpvec*exchrate
xpcec=xpcec/exchrate
endif
```

Step 10: **Consistency checks are performed between the *.dec file and the *.prd file of the reservoir performance module.**

Note: An error message is printed if there is an inconsistency.

```
if((rsty.lt.1).or.(rsty.gt.nrst)) call errmsg(4,451)

if(ii.eq.3) then
f=min0(max0(mxnfsz-(f1-4)+1,1),mxnfsz)
```

```
      endif

      if(ii.eq.3) then
      ctsto=ctst
      ipno=ipn
      rstyo=rsty
      f1o=f1
      endif
      if(ctsto.ne.ctst) write(*,9141) ipn,rsty,ctst,f1,
      ipno,rstyo,ctsto,f1o
9141  format(' ',2(a2,i1,a4,i3))
      if(ipno.ne.ipn)  write(*,9141) ipn,rsty,ctst,f1,
      ipno,rstyo,ctsto,f1o
      if(f1o.ne.f1)  write(*,9141) ipn,rsty,ctst,f1,
      ipno,rstyo,ctsto,f1o
      if(rstyo.ne.rsty)  write(*,9141) ipn,rsty,ctst,f1,
      ipno,rstyo,ctsto,f1o
      if(ctsto.ne.ctst) call errmsg(4,452)
      if(ipno.ne.ipn) call errmsg(4,452)
      if(f1o.ne.f1) call errmsg(4,452)
      if(rstyo.ne.rsty) call errmsg(4,452)
```

Step 11: **The play name is assigned into the variable 'pname' where 'pname' is a character which is 20 strings long.**

```
      pname='                    '
      pname(1:4)=ctst
```

Step 12: **Consistency checks are performed between the reservoir data bank file and the play definition file ('ply_dfn.spc').**

Note: An error message is printed if there is an inconsistency.

```
      if((po.ne.0).and.(pname.eq.pnameo)) then
      p=po
      else
      pnameo=pname
      do 1152 p=1,nply
      if(pname.eq.plynme(p)) go to 1153
1152  continue
      write(*,1151) pname,f
      call errmsg(4,453)
1153  continue
      endif
      if((f.le.0).or.(f.gt.mxnfsz)) then
      write(*,1151) pname,f
1151  format(' Play: ',a20,' Field Size: ',i2)
      call errmsg(4,454)
      endif
```

Step 13: **The resource types in the data bank file are overwritten by the play specific resource types from the 'ply_dfn.spc' file.**

```
rsty=rst(p)
```

Step 14: **A consistency check is performed to ensure that the value of the pay grade is valid.**

```
if((d.le.0).or.(d.gt.3)) then
write(*,1151) pname,f
call errmsg(4,455)
endif
```

Step 15: **The variables 'wi' and 'b' are assigned values based on technology types.**

```
if(ctch.eq.'C') then
wi=1
elseif(ctch.eq.'M') then
wi=2
elseif(ctch.eq.'A') then
wi=3
elseif(ctch.eq.'E') then
wi=3
else
write(*,1151) pname,f
call errmsg(4,456)
endif
if(copt.eq.'P') then
b=1
elseif(copt.eq.'R') then
b=2
elseif(copt.eq.'I') then
b=3
else
write(*,1151) pname,f
call errmsg(4,457)
endif
```

Step 16: **The validity of the variable representing the resource type (rsty) is verified.**

```
if((rsty.le.0).or.(rsty.gt.nrst)) call errmsg(4,458)
```

Step 17: **Reserves available for exploration by resource type and play are stored in 'etcrsq(rsty,p)'.**

```
etcrsq(rsty,p)=etcrsq(rsty,p)+xres
```

Step 18: **Here the two variables 'kdnwlb(v)' and 'kdnwlc(v)' are calculated.**

Note: 'Kdnwlb(v)' is number of undiscovered and reserve growth reservoirs in the FSC of the Play. 'Kdnwlc(v)' is the original number of undiscovered and reserve growth reservoirs in the FSC of the Play (this variable doesn't change with time). Both these variables are calculated using 'curbnk(f,igrsr)' which is the factor of the reserve growth relative to the undiscovered reservoirs in a given FSC in a region. 'Nrrr' is the number of undiscovered reservoirs.

```
kdnwlb(v)=nrrr*(1+curbnk(f,igrsr))
kdnwlc(v)=nrrr*(1+curbnk(f,igrsr))
```

Note: The general format followed in the following lines of code is:

a) Internal consistency checks are performed to make sure that the data is consistent, if there is an inconsistency error messages are printed.
b) The data is re-processed and stored in an acceptable format.

Step 19: **The validity of the variable 'nnfl' is verified and the arrays 'nkpnt' and 'kpnt' are set up which map one reservoir pointer (nnfl) to another (v) with the same value.**

```
        if (nnflp(f,p).eq.0) then
        nnfl=nnfl+1               ! nnflp is no of reservoirs to be processed
        if(nnfl.gt.mxnnfl) call errmsg(4,459)
        g=nnfl
        nnflp(f,p)=g
        nfsz(p)=max0(nfsz(p),f)      ! p is play counter inply_dfn.spc file
        nnflb(nnfl)=p*mxnfsz+f-1
        nkpnt(g)=v
        nnflx(g)=0.0
        else
        g=nnflp(f,p)
        v1=nkpnt(g)
1150    if(kpnt(v1).eq.0) go to 1155
        v1=kpnt(v1)
        go to 1150
1155    if(v1.ne.v) then
        kpnt(v1)=v
        endif
        endif
```

Step 20: The variables below are adjusted for the number of reservoirs (nrrr) and the values are stored.

```
        if(ii.eq.3) then  !checks to see whether this is the first reservoir.
        kdnwlb(v)=nrrr*(1+curbnk(f,igrsr))
        kdnwlc(v)=nrrr*(1+curbnk(f,igrsr))
        nnflx(g)=nnflx(g)+nrrr
        plynfl(f,p)=plynfl(f,p)+nrrr                !no. of undiscovered reservoirs in field
        plybfl(f,p)=0.0                     !No. of disc. undeveloped reservoirs in FSC in play, initially ZERO
        plybfl_rg(f,p)=plybfl_rg(f,p)+nrrr*curbnk(f,igrsr) !No. of reservoirs available for RES.GROWTH over the
        entire run
```

Step 21: The value of 'nnflpo' is assigned using 'kdnwlc' and the initial value of 'nnflpr' is assigned.

Note: The values of 'kdrsty' and 'ndrsty' are also assigned.

```
        nnflpo(f,p)=nnflpo(f,p)+kdnwlc(v)
        nnflpr(f,p)=nnflpo(f,p)
        endif

        kdrsty(v)=rsty
        if(ndrsty(g).eq.0) then
        ndrsty(g)=rsty
        endif
```

Step 22: The depth and window variables below are verified and adjusted for the number of reservoirs (nrrr) and their value is stored.

```
        if(ii.eq.3) then
        nddpth(g)=nddpth(g)+idpth*nrrr             !total footage needed to be drilled for FSC in the play
        endif

        if(kddpth(v).eq.0) then
        kddpth(v)=idpth
        elseif(kddpth(v).ne.idpth) then
        write(*,9611) ipn,ctst,idpth,kddpth(v)
9611    format(' ',a2,a4,2f12.1)
        write(*,1151) pname,f
        call errmsg(3,461)
        endif
        if(ii.eq.3) then
```

```
ndwin(wi,g)=ndwin(wi,g)+iwin*nrrr
endif
if(kdwin(wi,v).eq.0) then
kdwin(wi,v)=iwin
elseif(d.eq.2) then
kdwin(wi,v)=iwin
endif
```

Step 23: **The number of wells for infill drilling is hardwired to be twice the number of wells in the reservoir, therefore here the actual number of wells in the reservoir is half the number of wells in the database if there is infill drilling.**

Note: Here the code reflects this since if 'b' equals three then there is infill drilling, and 'xnw' which is the number of wells is reset to 'xnw/2'.

```
if(b.eq.3) then
xnw=xnw/2.0
endif
if((b.eq.1).and.(wi.eq.1)) then
ndnwl(d,g)=ndnwl(d,g)+xnw*nrrr
endif
```

Step 24: **Here the variables representing the number of wells in a discovered reservoir are initialized.**

Note: kdnwl(1,d,v) represents the total number of discovered wells, kdnwl(2,d,v) represents the primary producing wells, kdnwl(3,d,v) represents the secondary producing wells, and kdnwl(4,d,v) represents the undiscovered wells

```
if(kdnwl(1,d,v).eq.0) then   !total producing wells since undisc.
kdnwl(1,d,v)=0.0
kdnwl(2,d,v)=0.0
kdnwl(3,d,v)=0.0
kdnwl(4,d,v)=xnw           !total no. wells.
kdnwla(d,v)=0.0
elseif(kdnwl(4,d,v).ne.xnw) then
val=kdnwl(4,d,v)-xnw
if((val.lt.-0.5).or.(val.gt.0.5)) then
write(*,1151) pname,f
call errmsg(4,463)
endif
endif
```

Step 25: **The productive life of a well is assigned to the variable kdpryr.**

```
if(ii.eq.3) then
kdpryr(b,d,wi,g)=ipryr    ! g will go only upto undiscovered
endif
if(kdpryr(b,d,wi,v).eq.0) kdpryr(b,d,wi,v)=ipryr  !v is total counter
```

Step 26: **The validity of the present value of production for all the discovered reservoirs in a FSC of the play (kdpvp(b,d,wi,v)) is verified.**

Note: The value of the present value of production for all the undiscovered reservoirs in an FSC of the play (ndpvp(b,d,wi,g)) is re-set to zero.

```
if(kdpvp(b,d,wi,v).ne.-1.0) call errmsg(4,464)
if(ndpvp(b,d,wi,g).eq.-1.0) ndpvp(b,d,wi,g)=0.0
```

Step 27: **The variables below are adjusted for the number of reservoirs (nrrr) and their value is stored.**

Note: Increase in NPV of non-drilling cost for increase in gas price of $1.00 (ndpcnd(b,d,wi,g)) is set equal to changes in investment as the price of gas goes from $5 to $2 normalized per dollar (xpctc/3) times the ratio of the NPV of non-drilling cost (xpvndc) and the total NPV for non-drilling as well as drilling cost (xpvndc+xpvdc). This quantity is then adjusted for the number of reservoirs (nrrr). The equivalent equation is used for the increase in NPV of drilling cost for increase in gas price of $1.00 (ndpcdc(b,d,wi,g))

```
ndpvp(b,d,wi,g) =ndpvp(b,d,wi,g) +xpvp*nrrr
ndpvnd(b,d,wi,g)=ndpvnd(b,d,wi,g)+(xpvndc+xpvec)*nrrr
ndpvdc(b,d,wi,g)=ndpvdc(b,d,wi,g)+xpvdc*nrrr
ndpvtx(b,d,wi,g)=ndpvtx(b,d,wi,g)+ xpvtax*nrrr
ndpcnd(b,d,wi,g)=ndpcnd(b,d,wi,g)+xpcec/3.00*nrrr   !3.00 is $5-$2 = 3.00, RP drives the difference for 3
dollars/mcf

if(xpvndc.gt.0.0) then                          !non-drilling costs
ndpcnd(b,d,wi,g)=ndpcnd(b,d,wi,g)+xpctc*(xpvndc/(xpvndc+xpvdc))/
3.00*nrrr
endif
if(xpvdc.gt.0.0) then
ndpcdc(b,d,wi,g)=ndpcdc(b,d,wi,g)+xpctc*(xpvdc/(xpvdc+xpvndc))/
3.00*nrrr
endif
ndpctx(b,d,wi,g)=ndpctx(b,d,wi,g)+xpctax/3.00*nrrr
ndpvds(b,d,wi,g)=ndpvds(b,d,wi,g)+xpvds*nrrr
ndpvns(b,d,wi,g)=ndpvns(b,d,wi,g)+xpvnds*nrrr
ndpryr(b,d,wi,g)=ipryr
if((b.eq.1).and.(wi.eq.1)) then
ndogip(d,g)=ndogip(d,g)+xogip*nrrr
```

```
        endif
```

Step 28: **The variables below (which have been read from news\undb.bnk') are stored in the relevant arrays (for example: 'xpvp' in 'kdpvp(b,d,wi,v)').**

Note: Increase in NPV of non-drilling cost for increase in gas price of $1.00 for known reservoirs (kdpcnd(b,d,wi,g)) is set equal to a) changes in investment as the price of gas goes from $5 to $2 normalized per dollar (xpctc/3) times the ratio of the NPV of non-drilling cost (xpvndc) and the total NPV for non-drilling as well as drilling cost (xpvndc+xpvdc); b) this quantity reflecting investment costs is then added to the equivalent quantity representing change in expenses per dollar (xpcec/3.00). The equivalent equation for increase in NPV of drilling cost for increase in gas price of $1.00 for known reservoirs (kdpcdc(b,d,wi,g)) is set equal to total costs minus non-drilling cost (kdpcnd(b,d,wi,g)).

```
        kdpvp(b,d,wi,v)=xpvp
        kdpvnd(b,d,wi,v)=xpvndc+xpvec
        kdpvdc(b,d,wi,v)=xpvdc
        kdpvtx(b,d,wi,v)=xpvtax
        kdpryr(b,d,wi,v)=ipryr
        if(xpvndc.gt.0.0) then
        kdpcnd(b,d,wi,v)=xpctc*(xpvndc/(xpvndc+xpvdc))/3.00+xpcec/3.00
        else
        kdpcnd(b,d,wi,v)=xpcec/3.00
        endif
        kdpcdc(b,d,wi,v)=xpctc/3.00+xpcec/3.0-kdpcnd(b,d,wi,v)
        kdpctx(b,d,wi,v)=xpctax/3.00
        kdpvds(b,d,wi,v)=xpvds
        kdpvns(b,d,wi,v)=xpvnds
        kdfldt(v)=xdeqy(1,1,rsty)
        kdflpp(v)=p
        kdrsty(v)=rsty
        kdogip(d,v)=xogip
        go to 1130

1210    close(24)
```

Step 29: **Internal consistency checks are performed to make sure that the data is consistent.**

Note: If there is an inconsistency error messages are printed.

```
        v=v-1

        do 1230 p=1,nply
        rsty=1
        do 1215 i=1,nrst
        if(etcrsq(i,p).gt.etcrsq(rsty,p)) then
        rsty=i
        endif
1215    continue
        plyrst(p)=rsty
        j=0
1230    continue
        if((nnfl+1).gt.mxnnfl) call errmsg(4,465)
        do 1250 p=1,nply
        if(nfsz(p).gt.0) then
        do 1240 f=1,nfsz(p)
        if(nnflp(f,p).eq.0) nnflp(f,p)=nnfl+1
1240    continue
        endif
1250    continue
        write(*,1251) nnfl
1251    format(' Number of Undiscovered Reservoirs: ',i4)
```

Step 30: **The variable 'wi' represents technology type. The value of 'wi' = 2 will never be used because the value for 'wi' for current technology is 1 and for advanced technology is 2.**

Note: Here it is ensured that in case of an error (i.e., when 'wi' equals 2) the value of the variables is set to the current technology values.

```
        do 1270 g=1,nnfl
        do 1265 wi=1,mxndty  !technology C/A
        do 1260 d=1,mxngr
        do 1255 b=1,3      !dev. type P/R/I
        if(ndpvp(b,d,wi,g).eq.-1.0) then
        if(wi.eq.2) then
        ndpvp(b,d,wi,g)= ndpvp(b,d,1,g)
        ndpvnd(b,d,wi,g)=ndpvnd(b,d,1,g)
        ndpvdc(b,d,wi,g)=ndpvdc(b,d,1,g)
        ndpvtx(b,d,wi,g)=ndpvtx(b,d,1,g)
        ndpcnd(b,d,wi,g)=ndpcnd(b,d,1,g)
        ndpcdc(b,d,wi,g)=ndpcdc(b,d,1,g)
        ndpctx(b,d,wi,g)=ndpctx(b,d,1,g)
        ndpvds(b,d,wi,g)=ndpvds(b,d,1,g)
        ndpvns(b,d,wi,g)=ndpvns(b,d,1,g)
        else
        call errmsg(4,466)
        endif
        endif
1255    continue
1260    continue
1265    continue
1270    continue
```

Step 31: **In the above lines of code, these variables were multiplied with 'nrrr' which was assigned the value of the variable 'nnflx', here we are simply dividing the variables (discussed above) with**

nnflx, as a result there is no change in the value of these variables.

```
      do 1290 g=1,nnfl
      if(nnflx(g).gt.0.0) then
      nddpth(g)=nddpth(g)/nnflx(g)
      do 1285 wi=1,mxndty
      ndwin(wi,g)=ndwin(wi,g)/nnflx(g)
      do 1280 d=1,mxngr
      if(wi.eq.1) then
      ndnwl(d,g)=ndnwl(d,g)/nnflx(g)
      endif
      do 1275 b=1,3
      ndpvp(b,d,wi,g)= ndpvp(b,d,wi,g) /nnflx(g)
      ndpvnd(b,d,wi,g)=ndpvnd(b,d,wi,g)/ nnflx(g)
      ndpvdc(b,d,wi,g)=ndpvdc(b,d,wi,g)/nnflx(g)
      ndpvtx(b,d,wi,g)=ndpvtx(b,d,wi,g)/nnflx(g)
      ndpcnd(b,d,wi,g)=ndpcnd(b,d,wi,g)/nnflx(g)
      ndpcdc(b,d,wi,g)=ndpcdc(b,d,wi,g)/nnflx(g)
      ndpctx(b,d,wi,g)=ndpctx(b,d,wi,g)/nnflx(g)
      ndpvds(b,d,wi,g)=ndpvds(b,d,wi,g)/nnflx(g)
      ndpvns(b,d,wi,g)=ndpvns(b,d,wi,g)/nnflx(g)
      if((b.eq.1).and.(wi.eq.1)) then
      ndogip(d,g)=ndogip(d,g)/nnflx(g)
      endif
1275  continue
1280  continue
1285  continue
      endif
1290  continue
      call gett(tmes(6),tmea(6),1)
      call errmsg(1,914)
```

Step 32: **The discovered data bank file 'news\disb.bnk' is opened.**

Note: The two header lines are read and stored in the dummy variable 'ctch'. The following variables are initialized:

1) Pointers (such as 'nuds','nefl','ctsto','ii').
2) Reservoir type for discovered reservoirs (kdrsty(v))
3) Total footage needed to be drilled for all discovered reservoirs in FSC of the play (kddpth(v))
4) For a discovered reservoir the year that window opens (relative to the starting year of production) (kdwin(wi,v))
5) Number of wells in a discovered reservoir (kdnwl(1,d,v))
6) Present value of production for all the discovered reservoirs in FSC of the play (kdpvp(b,d,wi,v))
7) A pointer for new and undiscovered reservoirs to the next one in the same FSC (kpnt(v))

```
      nuds=0
```

```
        call gett(tmes(7),tmea(7),0)
1300    if(iocde.eq.0) then
        open(25,file='news\dis.bnk')
        open(35,file='news\disb.bnk',form='BINARY')
        else
        open(25,file='news\disb.bnk',form='BINARY')
        endif
        nefl=v
        do 1305 v=nefl+1,mxnefl
        kdrsty(v)=0
        kddpth(v)=0
        kpnt(v)=0
        do 1304 wi=1,mxndty
        kdwin(wi,v)=0
        do 1303 d=1,mxngr
        kdnwl(1,d,v)=0
        do 1302 b=1,mxdevt
        kdpvp(b,d,wi,v)=-1.0
1302    continue
1303    continue
1304    continue
1305    continue
        ii=0
1310    ii=ii+1
        if(ii.le.2) then
        if(iocde.eq.0) read(25,1129,end=1350) ctch
        if(iocde.eq.0) read(25,1129,end=1350) ctch
        ii=ii+2
        ctsto='    '
        elseif(ii.eq.20) then
        ii=0
        endif
```

Step 33: **The Development Profiles are read from the 'news\undb.bnk' file by Play, Field Size, and Technology.**

Note: The following variables are read:

1) Supply region code (ipn)
2) Status of resource (c1)
3) Resource Type (rsty)
4) USGS Play number (ctst)
5) Field size class of a GSAM ID for an undiscovered reservoir (f1)
6) Technology case (ctch)
7) Paygrade (d)
8) Development type case (Primary, Refrac., Infill), (copt)
9) Technically recoverable reserves (Bcf), (xres)
10) Original gas in place (xogip)
11) Number of wells (xnw)
12) MASP (xmasp)
13) Total capacity (xtcap)
14) NPV of production in Bcf (xpvp)

15) NPV of expenses in million $ (xpvec)
16) NPV of investment in million $ (xpvtc)
17) NPV of drilling cost in million $ (xpvdc)
18) NPV of non-drilling cost in million $ (xpvndc)
19) Severance, federal and state taxes in million $ (xpvtax)
20) Changes in expenses as the price of gas goes from $5 to $2 (ratio), (xpcec)
21) Changes in investment as the price of gas goes from $5 to $2 (ratio), (xpctc)
22) Changes in taxes as the price of gas goes from $5 to $2 (ratio), (xpctax)
23) Drill slope (xpvds)
24) Non-drilling slope (xpvnds)
25) Reservoir depth (idpth)
26) Water depth (ih2o)
27) Window year (iwin)
28) Productive life (ipryr)
29) Number of reservoirs (nrrr)

```
read(25,end=1350)    ipn,c1,rsty,ctst,f1,ctch,d,copt,xres,xogip,
xnw,xmasp,xtcap,xpvp,xpvec,xpvtc,xpvdc,xpvndc,xpvtax,
xpcec,xpctc,xpctax,xpvds,xpvnds,idpth,ih2o,iwin,
ipryr
```

Step 34: **The following variables are stored in such a way that the production variables are in MMSCF and cost variables are in thousand dollars.**

```
if(iwin.lt.2) iwin=2
xres=xres*1000.0
xogip=xogip*1000.0
xtcap=xtcap*1000.0
xpvp=xpvp*1000.0
xpvec=xpvec*1000.0
xpvtc=xpvtc*1000.0
xpvdc=xpvdc*1000.0
xpvndc=xpvndc*1000.0
xpvtax=xpvtax*1000.0
xpcec=xpcec*1000.0
xpctc=xpctc*1000.0
xpctax=xpctax*1000.0
```

Step 35: **The expense variables for Canadian reservoirs are adjusted using the exchange rate.**

```
if(ipn.eq.'22'.or.ipn.eq.'23'.or.ipn.eq.'24')then
```

```
        xpvec=xpvec*exchrate
        xpcec=xpcec/exchrate
        endif
```

Step 36: **Consistency checks are performed between the *.dec file and the *.prd file of the reservoir performance module.**

Note: An error message is printed if there is an inconsistency.

```
        f=f1
        if(ii.eq.3) then
        ctsto=ctst
        ipno=ipn
        rstyo=rsty
        f1o=f1
        endif
        if(ctsto.ne.ctst) call errmsg(4,467)
        if(ipno.ne.ipn) call errmsg(4,467)
        if(rstyo.ne.rsty) call errmsg(4,467)
        if(f1o.ne.f1) call errmsg(4,467)
```

Step 37: **The play name is assigned into the variable 'pname' where 'pname' is a character which is 20 strings long.**

```
        pname='                    '
        pname(1:4)=ctst
```

Step 38: **Consistency checks are performed between the reservoir data bank file and the play definition file ('ply_dfn.spc').**

Note: An error message is printed if there is an inconsistency.

```
        if((po.ne.0).and.(pname.eq.pnameo)) then
        p=po
        else
        pnameo=pname
        do 1315 p=1,nply
        if(pname.eq.plynme(p)) go to 1320
1315    continue
        write(*,1316) pname
1316    format(' play: ',a20)
        call errmsg(4,468)
1320    continue
        endif
```

Step 39: **The resource types in the data bank file are overwritten by the play specific resource types from the 'ply_dfn.spc' file.**

```
rsty=rst(p)
```

Step 40: **The number of reservoirs still left to be discovered (kdnwlb(v)) and the original number of reservoirs left to be discovered for the specific reservoir type (kdnwlc(v)) are initialized.**

```
if(ii.eq.3) then
nefl=nefl+1
v=nefl
kpnt(v)=0
if(nefl.gt.mxnefl) call errmsg(4,469)
vo=v
kdnwlb(v)=0.0
kdnwlc(v)=0.0
else
v=vo
endif
```

Step 41: **A consistency check is performed to ensure that the value of the pay grade is valid.**

```
if((d.le.0).or.(d.gt.3)) then
write(*,1151) pname,f
call errmsg(4,470)
endif
```

Step 42: **The validity of the variables representing the resource type (rsty) is verified.**

```
if((rsty.le.0).or.(rsty.gt.nrst)) call errmsg(4,471)
```

Step 43: **The variables 'wi' and 'b' are assigned values based on technology types. In addition ‘resource-technology’ combinations are assigned and play counters are stored.**

```
1335    kdfldt(v)=xdeqy(1,1,rsty)
        if(kflag.eq.'ADV') then
        if(xdeqy(1,3,rsty).ne.0) then
        kdfldt(v)=xdeqy(1,3,rsty)
        endif
```

```
endif
kdflpp(v)=p


if(ctch.eq.'C') then
wi=1
elseif(ctch.eq.'M') then
wi=2
elseif(ctch.eq.'A') then
wi=3
elseif(ctch.eq.'E') then
wi=3
else
write(*,1151) pname,f
call errmsg(4,472)
endif

if(copt.eq.'P') then
b=1
elseif(copt.eq.'R') then
b=2
elseif(copt.eq.'I') then
b=3
else
write(*,1151) pname,f
call errmsg(4,473)
endif
```

Step 44: The resource type (kdrsty(v)), depth (kddpth(v)), and window variables (kdwin(wi,v)) below are verified and their value is stored.

```
if(kdrsty(v).eq.0) then
kdrsty(v)=rsty
elseif(kdrsty(v).ne.rsty) then
write(*,1151) pname,f
call errmsg(4,474)
endif

if(kddpth(v).eq.0) then
kddpth(v)=idpth
elseif(kddpth(v).ne.idpth) then
write(*,1151) pname,f
call errmsg(4,475)
endif

if(kdwin(wi,v).eq.0) then
kdwin(wi,v)=iwin
elseif(d.eq.2) then
kdwin(wi,v)=iwin
endif
```

Step 45: The number of wells for infill drilling is hardwired to be twice the number of wells in the reservoir, therefore here the actual

number of wells in the reservoir is half the number of wells in the database if there is infill drilling.

Note: Here the code reflects this since if 'b' equals three then there is infill drilling, and 'xnw' which is the number of wells is reset to 'xnw/2'.

```
if(b.eq.3) then
xnw=xnw/2.0
endif
```

Step 46: **Here the variables representing the number of wells in a discovered reservoir are initialized.**

Note: kdnwl(1,d,v) represents the total number of discovered wells
kdnwl(2,d,v) represents the primary producing wells
kdnwl(3,d,v) represents the secondary producing wells
kdnwl(4,d,v) represents the undiscovered wells

```
if(kdnwl(1,d,v).eq.0) then
kdnwl(1,d,v)=xnw
if(c1.ne.'2') then
kdnwl(2,d,v)=xnw
else
kdnwl(2,d,v)=0.0
endif
kdnwl(3,d,v)=0.0
kdnwl(4,d,v)=xnw
kdnwla(d,v)=kdnwl(1,d,v)-kdnwl(2,d,v)
elseif((kdnwl(4,d,v).ne.xnw).and.(xnw.ne.0.0)) then
val=kdnwl(4,d,v)-xnw
if((val.lt.-0.5).or.(val.gt.0.5)) then
endif
endif
```

Step 47: **The validity of the present value of production for all the discovered reservoirs in a FSC of the play (kdpvp(b,d,wi,v)) is verified.**

```
if(kdpvp(b,d,wi,v).ne.-1.0) call errmsg(4,478)
```

Step 48: **The variables (which have been read from 'news\disb.bnk') are stored in the relevant arrays (for example: 'xpvp' in 'kdpvp(b,d,wi,v)').**

Note: Increase in NPV of non-drilling cost for increase in gas price of $1.00 for known reservoirs (kdpcnd(b,d,wi,g)) is set equal to a) changes in investment as the price of gas goes from $5 to $2 normalized per dollar (xpctc/3) times the ratio of the NPV of non-drilling cost (xpvndc) and the total NPV for non-drilling as well as drilling cost (xpvndc+xpvdc); b) this quantity reflecting investment costs is then added to the equivalent quantity representing change in expenses per dollar (xpcec/3.00). The equivalent equation for increase in NPV of drilling cost for increase in gas price of $1.00 for known reservoirs (kdpcdc(b,d,wi,g)) is set equal to total costs minus non-drilling cost (kdpcnd(b,d,wi,g)).

```
        kdpvp(b,d,wi,v)=xpvp
        kdpvnd(b,d,wi,v)=xpvndc+xpvec
        kdpvdc(b,d,wi,v)=xpvdc
        kdpvtx(b,d,wi,v)=xpvtax
        if(xpvndc.gt.0.0) then
        kdpcnd(b,d,wi,v)=xpctc*(xpvndc/(xpvndc+xpvdc))/3.00+xpcec/3.00
        else
        kdpcnd(b,d,wi,v)=xpcec/3.00
        endif
        kdpcdc(b,d,wi,v)=xpctc/3.00+xpcec/3.00-kdpcnd(b,d,wi,v)
        kdpctx(b,d,wi,v)=xpctax/3.00
        kdpvds(b,d,wi,v)=xpvds
        kdpvns(b,d,wi,v)=xpvnds
        kdpryr(b,d,wi,v)=ipryr
        kdogip(d,v)=xogip
        go to 1310

1350    close(25)
```

Step 49: **Internal consistency checks are performed to make sure that the data is consistent. If there is an inconsistency, error messages are printed.**

```
        if(iocde.eq.0) then
        endfile 35
        close(35)
        endif
        if(nefl.le.0) call errmsg(4,479)
```

Step 50: **The variable 'wi' represents technology type. The value of 'wi' = 2 will never be used because the value for 'wi' for current technology is 1 and for advanced technology is 2.**

Note: Here it is ensured that in case of an error (i.e., when 'wi' equals 2) the value of the variables are assigned as the current technology values.

```
        do 1370 v=1,nefl
        do 1365 wi=1,3
        do 1360 d=1,mxngr
        do 1355 b=1,3
        if(kdpvp(b,d,wi,v).eq.-1.0) then
        if((wi.eq.2).or.
        ((wi.eq.3).and.(kdnwl(1,d,v).le.kdnwl(2,d,v)))) then
        kdpvp(b,d,wi,v)= kdpvp(b,d,1,v)
        kdpvnd(b,d,wi,v)=kdpvnd(b,d,1,v)
        kdpvdc(b,d,wi,v)=kdpvdc(b,d,1,v)
        kdpvtx(b,d,wi,v)=kdpvtx(b,d,1,v)
        kdpcnd(b,d,wi,v)=kdpcnd(b,d,1,v)
        kdpcdc(b,d,wi,v)=kdpcdc(b,d,1,v)
        kdpctx(b,d,wi,v)=kdpctx(b,d,1,v)
        kdpvds(b,d,wi,v)=kdpvds(b,d,1,v)
        kdpvns(b,d,wi,v)=kdpvns(b,d,1,v)
        if(kdwin(wi,v).eq.0) kdwin(wi,v)=kdwin(1,v)
        else
        call errmsg(4,480)
        endif
        endif
1355    continue
1360    continue
1365    continue
1370 c  ontinue
```

Step 51: **Here a number of pointers and back pointers are set up to be used in later subroutines.**

Note: The variable 'jwls' and 'jwlp' represent the sum of wells by pay grade that are secondary wells and the additional wells that need to be drilled, respectively. 'nuds' is a pointer for the number of reservoirs, 'upb' is a back pointer, 'upf' is a front pointer.

```
        do 1490 v=1,nefl
        y1=1
        jwls=0
        jwlp=0
        do 1430 d=1,mxngr
        jwls=jwls+kdnwl(3,d,v)
        jwlp=jwlp+kdnwl(2,d,v)-kdnwl(3,d,v)
1430    continue
```

Step 52: **This condition of 'jwls' greater than zero is never satisfied since 'kdnwl(3,d,v)' is always equal to zero by definition.**

```
        if(jwls.gt.0.0) then
        nuds=nuds+1
        if(nuds.gt.mxnuds) call errmsg(4,481)
```

```
          if(nuds.gt.1) then
          upb(nuds)=nuds-1
          upf(nuds)=0
          upf(nuds-1)=nuds
          else
          upb(nuds)=0
          upf(nuds)=0
          endif
          upcde(nuds)=v
          uptchp(nuds)=kdfldt(v)
          uptchs(nuds)=kdfldt(v)
          upsect(nuds)=2
          jwlp=0
          upyr(nuds)=tmex-y1
          upsyr(nuds)=upyr(nuds)+kdwin(1,v)
          endif
```

Step 53: **Here 'uptchp' and 'uptchs' which represent the index of technology of primary and secondary development respectively are assigned values.**

Note: The value for secondary development is set at zero because we never have secondary production in a discovered field. The number of wells by pay grade in a project ('upnwl') is set to 'kdnwl(2,d,v)' since 'kdnwl(3,d,v)' is zero by definition.

```
          if(jwlp.gt.0) then
          nuds=nuds+1
          if(nuds.gt.mxnuds) call errmsg(4,482)
          if(nuds.gt.1) then
          upb(nuds)=nuds-1
          upf(nuds)=0
          upf(nuds-1)=nuds
          else
          upb(nuds)=0
          upf(nuds)=0
          endif
          upcde(nuds)=v
          uptchp(nuds)=kdfldt(v)
          uptchs(nuds)=0
          do 1440 d=1,mxngr
          upnwl(d,nuds)=kdnwl(2,d,v)-kdnwl(3,d,v)
1440      continue
```

Step 54: **The variables ‘upyr’ and ‘upsyr’ are set up here to be used in later subroutines.**

Note: 'upyr' represents the starting year of development program, 'upsyr' represents the starting year of secondary development. Here 'upsyr' is set to zero since there is no secondary development.

```
        upyr(nuds)=tmex-y1
        upsyr(nuds)=0
        endif

1490    continue
```

Step 55: These pointers ('upfrst' and 'uplast') are set up here to be used in later subroutines.

```
        call gett(tmes(7),tmea(7),1)
        if(nuds.le.0) call errmsg(3,483)
        upfrst=1
        uplast=nuds
        call errmsg(1,915)
        write(*,1391) nefl
1391    format(' Total Reservoir Count: ',i5)

        return
        end
```

Table of Contents

SUBROUTINE ENV_READ

CALLED BY: EXPLPROD (Controls the main flow through the expl./dvlp. model.)

CALLS: SET2ZERO (This initializes all the environmental variables to zero.)
NZEROVAL (Sets environmental variables to a temporary variable 'val'.)

READS: 'env_stat.spc' (Contains static cost data.)
'env_proc.spc' (Contains processing cost data.)
'env_dat.spc' (Contains the number and names of the environmental files.)

MAIN THEME: This routine reads in the environmental and processing cost data from the 'env_stat.spc' file, the 'env_proc.spc' file and the files specified in **'**env_dat.spc' .

Step 1: **The input files are opened.**

Note: The input file 'env_stat.spc' contains location data, impurity levels, federal functions, etc. and the input file 'env_proc.spc' contains effective gas processing cost data in $/MCF. The input file 'env_dat.spc' contains information on the number and the name of the environmental files created from GSAM's environmental module. The common index for each of these files is the GSAMID. The variable 'ienvcse' is the total number of environmental regulations to be read. The environmental regulations are specified on a yearly basis.

```
open(unit=51,file='env_stat.spc')
open(unit=52,file='env_proc.spc')
open(53,file='env_dat.spc')

read(53,*)ienvcse, vscl
```

Step 2: **The 'env_dat.spc' file is read.**

Note: The year (iyrenv) for which the environmental regulations are applicable and the name of environmental files (envfile(ienv)) are read from 'env_dat.spc'. A default value for the year for the inadverdant case after the final case (ienvcse+1) is also assigned which is larger than the timeframe to be modeled. Once the information is read the 'env_dat.spc' file is closed.

```
        do ienv=1,ienvcse
        read(53,900)iyrenv(ienv),envfile(ienv)
        enddo
        iyrenv(ienvcse+1)=tme(ntme)+1
        close(53)
```

Step 3: **The 'env_proc.spc' file is read.**

Note: At first the processing cost information is read by looping over for all the discovered and undiscovered reservoirs. The following variables are read:

1) GSAMID which is equivalent to id2 in the following step.
2) Processing cost in $/MCF (proc_cst(1,iv))

```
        Do iv = 1,nefl
        read(52,907,end=100) id2,proc_cst(1,iv)
907     format(a11,1x,f12.4)
905     format(a11,1x,i4,1x,f12.4,1x,f14.4,1x,f12.4,1x,f7.3,1x,f12.4,
1       1x,f12.4,1x,f12.4,1x,f12.4,1x,f12.4,1x,f12.4)
        Enddo
```

Step 4: **Variables are initialized.**

Note: Here the processing cost is set to zero for all remaining reservoirs for which gas processing cost was not specified. The following do loop starts from 'iv', i.e., from the point at which the earlier loop ended. The gas processing file 'env_proc.spc' (unit 52) is closed.

```
100     do iab=iv,nefl
        proc_cst(1,iab)=0.0
        enddo
        close(52)
```

Step 5: **Each environmental file is opened (i.e., for every ienvcase).**

Note: All the environmental variables are set to zero for the inadverdant case first.

```
        do ienv=1,ienvcse
        open(unit=53,file=envfile(ienv))
```

```
do iv=1,nefl
IF (ienv.eq.1) THEN
call set2zero(ienvcse+1,iv)
```

Step 6: **Here all the reservoir specific data pertinent for environmental cost calculations are read.**

Note: The following variables are read:

1) GSAMID (id1)
2) 4-digit state code (state(iv))
3) Depth (ft.) (depth(iv))
4) Area (acres) (area(iv))
5) Royalty rate (fraction) (royrate(iv))
6) Reservoirs federal fraction (frac_fed(iv))
7) Carbon Dioxide content (fraction) (co2(iv))
8) Nitrogen content (fraction) (n2(iv))
9) Hydrogen Sulfide content (fraction) (h2s(iv))
10) Lease bonus ($/ft) (lsb(iv))
11) Condensate yield (BBL/MCF) (condyld(iv))
12) Water Yield (BBl/MCF) (watyld(iv))

This read statement is accessed only once, therefore, if there are multiple environmental regulations, these entries are not re-read all the time.

```
read(51,905,end=110) id1,state(iv),depth(iv),area(iv),
1          royrate(iv),frac_fed(iv),co2(iv),
2          n2(iv),h2s(iv),lsb(iv),condyld(iv),watyld(iv)
ENDIF
```

Step 7: **Here variables are initialized.**

Note: The 'XXDOE.Env' files are opened and the GSAMID is verified, the variables are initialized using the 'set2zero' subroutine, and read using the 'nzeroval' subroutine. If there is a mismatch between ‘env_stat.spc’ and ‘XXDOE.Env’ file in terms of sequence of reservoir then an error message is printed and the program terminated. The nonzero environmental entries are stored in appropriate units for later use in the ‘nzeroval’ routine.

```
call set2zero(ienv,iv)
read(53,910,end=110)gsamid,num
```

```
        if (ienv.eq.1) then
        if(id1.ne.gsamid) then
        print *, 'Mismatch Between Env. Static File &'
        print *, 'File Specified in ENV_DAT.SPC File'
        print *, ' '
        print *, ' Check GSAMID ', id1, ' in ENV_STAT.SPC file'
        print *, ' Check GSAMID ', gsamid, ' in ENV_DAT.SPC file'
        stop
        end if
        end if
        do ii=1,num
        read(53,915) outnum(ii),outval(ii)
        call nzeroval(outnum(ii),outval(ii),ienv,iv)
        enddo ! loop for ii
4551    format(1x,a11,2x,i4,2x,f7.0,2x,f8.0,2x,f5.3,2x,i4,2x,f7.5,2x,
        f7.5,2x,f7.5,2x,f5.1,2x,
        f11.3,2x,f5.1,2x,10(f12.4,2x))

        enddo ! loop for iv (number of reservoirs/fields)
        close(53)
```

Step 8: Zero values are assigned for all the remaining reservoirs for which ‘env_stat.spc’ entries do not exist.

```
110     do iab=iv,nefl
        watyld(iab)          = 0.0
        envndec(ienv,iab)    = 0.0
        envdcec(ienv,iab)    = 0.0
        envexec(ienv,iab)    = 0.0
        envndnc(ienv,iab)    = 0.0
        envdcnc(ienv,iab)    = 0.0
        envexnc(ienv,iab)    = 0.0
        envdcf(ienv,iab)     = 0.0
        envgc(ienv,iab)      = 0.0
        envwc(ienv,iab)      = 0.0
        enddo
        enddo ! loop for number of environmental files
```

Step 9: The subroutine ends.

```
900     format(i4,1x,a15)
910     format(a11,1x,i2)
915     format(i2,1x,f12.4)

        return
        end
```

Step 10: The ‘nzeroval’ routine assigns the value of the variable ‘val’ to all the environmental arrays.

```
        Subroutine nzeroval(num,val,ienv,iv)
        integer*4 num,ienv,iv
        real *4 val
        include 'ex_sze.cmn'
        include 'dv_rpr.cmn'
```

```
if (num.eq.1)  then
envndec(ienv,iv) = val

else if (num.eq.2) then
envdcec(ienv,iv) = val

else if (num.eq.3) then
envexec(ienv,iv)  = val

else if (num.eq.4) then
envndnc(ienv,iv) = val

else if (num.eq.5) then
envdcnc(ienv,iv) = val

else if (num.eq.6) then
envexnc(ienv,iv) = val

else if (num.eq.7) then
envdcf(ienv,iv)  = val

else if (num.eq.8) then
envgc(ienv,iv)   = val

else if (num.eq.9) then
envwc(ienv,iv)   = val

end if
return
end
```

Table of Contents

SUBROUTINE EXDVI4A

CALLED BY: EXPLPROD (Controls the main flow through the expl./dvlp. model.)

CALLS: SKIPBIN (Routine to skip lines in input file.)
ERRMSG (Prints out errors and warnings)

READS: **‘**undb.tcp’ (Undiscovered binary file.)

MAIN THEME: This routine calculates the percentage of the total production for undiscovered reservoirs on which any applicable royalty incentive would be available. The incentive is calculated only on the federal percentage of the undiscovered reservoir. The federal percent is obtained from the ‘env_stat.spc’ file. All undiscovered reservoirs in a play are assigned the percentage as specified in the ‘ply_dfn.spc’ file.

Step 1: **The undiscovered data bank file (undb.tcp) is opened.**

Note: Production rates (by paygrade, development type and technology type) by year are read from the file to determine the total production from the reservoir.

```
open(1234,file='news\undb.tcp',form='BINARY')
```

Step 2: **Variables are initialized.**

Note: The frac_npv (this is the fraction of NPV Production on which royalty reduction can be applied) value is first initialized to zero for all undiscovered reservoirs (nnfl), alternative development types (mxndevt), paygrades (mxngr), and technology options (mxndty).

```
DO iv = 1, nnfl
Do wi = 1,mxndty
Do d  = 1,mxngr
Do b  = 1,mxdevt
frac_npv(b,d,wi,iv) = 0.0
frac_p(b,d,wi,iv)   = 0.0
Enddo
Enddo
```

```
        Enddo
        ENDDO
```

Step 3: **The undiscovered data bank file is read (undb.tcp).**

Note: The ‘Do Loop’ 124, is for all undiscovered reservoirs available in the databank. ‘Prd(i)’ is production rates (BCF/year) for a reservoir assuming all wells are on production. ‘Oam(i)’ is operating cost by year.

```
        Do 124 iv = 1,nnfl
        ii = 1

        read(1234,end=1160) ipn,c1,rsty,ctst,f1,ctch,d,copt,imyr,
        (prd(i),i=1,imyr),(oam(i),i=1,imyr)
```

Step 4: **The reservoir is skipped if there is no federal percentage as indicated in the ‘env_stat.spc’ file.**

```
        If (frac_fed(iv).le.0.0)  then
        call skipbin(1234,17)
        goto 124
        endif
```

Step 5: **Variables ‘wi’ and ‘b’ are assigned.**

Note: The variables 'wi' (for technology type, e.g., current technology ‘c’, or advanced technology ‘a’) and 'b' (for each development type, e.g., primary ‘p’, refraced ‘r’ or infill ‘i’) are assigned values based on variables read from the data bank file.

```
1102    if(ctch.eq.'C') then
        wi=1
        elseif(ctch.eq.'M') then
        wi=2
        elseif(ctch.eq.'A') then
        wi=3
        elseif(ctch.eq.'E') then
        wi=3
        else
        call errmsg(4,495)
        endif

        if(copt.eq.'P') then
        b=1
        elseif(copt.eq.'R') then
        b=2
        elseif(copt.eq.'I') then
        b=3
        else
        call errmsg(4,496)
        endif
```

Step 6: **The value of 'frac_npv' is adjusted and 'iflag_marg' is set.**

Note: 'Iflag_marg' is a counter which ensures that once an incentive is on it is always on. In this section of the code the value of 'frac_npv' (the fraction of production on which royalty incentive is applicable) is adjusted. The 'iflag_marg' variable is set based on the comparison of production rate per well in MCF/D/Well to the cutoff rate (rate_marg) as specified in the 'gen_tml.spc' file.

```
        iflag_marg = 0

        Do 1140 i=1,mxnyr
        IF (i.le.imyr.and.kdnwl(4,d,iv).gt.0.0) THEN
        If (prd(i)*1e06/(365*kdnwl(4,d,iv)).le.rate_marg .OR.
        iflag_marg.eq.1) Then
        iflag_marg = 1


        frac_npv(b,d,wi,iv) = frac_npv(b,d,wi,iv) +
        prd(i)/(1.0 + disrte/100.0)**(i-1)
        endif
        ENDIF
1140    Continue
```

Step 7: **'Frac_p' is calculated.**

Note: Here the value of 'frac_p' (fraction of NPV Production on which royalty reduction can be applied to the Total NPV Production) is calculated. The conversion factor of 1000.0 is used to convert the production value 'frac_npv' from BCF to MMCF.

```
        if (kdpvp(b,d,wi,iv).gt.0.0) then
        frac_p(b,d,wi,iv) = 1000.0*frac_npv(b,d,wi,iv)/ kdpvp(b,d,wi,iv)
        if (frac_p(b,d,wi,iv).gt.1.0) frac_p(b,d,wi,iv) = 1.0
        else
        frac_p(b,d,wi,iv) = 0.0
        endif
```

Step 8: **Here all the entries are read (18 in total: 3 paygrades, 3 development types and 2 technology options) for a reservoir and processed for 'frac_p' calculation.**

```
        ii = ii + 1
        if (ii.le.18) then
        read(1234,end=1160) ipn,c1,rsty,ctst,f1,ctch,d,copt,imyr,
        (prd(i),i=1,imyr),(oam(i),i=1,imyr)
        goto 1102
        endif
 124    continue
```

Step 9: The subroutine ends.

```
1160    close (1234)
        return
        end
```

Table of Contents

SUBROUTINE EXDVI4

CALLED BY: EXDVSO (The output is calculated and printed in output files.)

CALLS: GETT (Estimates the time required for each phase of the subroutine.)
ERRMSG (Prints out errors and warnings)

READS: ‘undb.tcp’ (Undiscovered binary file.)
‘disb.tcp’ (Discovered binary file.)

MAIN THEME: This routine reads the data files from the binary data bank files. It stores operating costs and production rates for all reservoirs for all development types, paygrades and technology options.

Step 1: The undiscovered and discovered bank files are opened.

Note: The flags showing the combinations that have been read in, are initialized.

```
1100    if(vi.eq.0) then
        open(24,file='news\undb.tcp',form='BINARY')
        open(25,file='news\disb.tcp',form='BINARY')
        vpl=0
        po=0
        pnameo='                    '
        ifl=24
        endif

        if(vi.lt.vpl) then
        rewind 24
        rewind 25
        vpl=0
        ifl=24
        endif
```

Step 2: The undiscovered reservoir data bank file is read. The production rate and operating and maintenance costs are read.

```
        if((vpl+1).lt.vi) then
        do 1110 vj=vpl+1,vi-1
        do 1105 ii=1,18
1102    read(ifl,end=1103)    ipn,c1,rsty,ctst,f1,ctch,d,copt,imyr,
        (prd(i),i=1,imyr),(oam(i),i=1,imyr)
        go to 1105
1103    if(ifl.ne.24) call errmsg(4,601)
        ifl=25
        go to 1102
1105    continue
1110    continue
        vpl=(vi-1)
        endif
```

Step 3: **The variable 'wprd' (indicating production rate) is initialized.**

```
        if(vpl.lt.vi) then
        do 1120 wi=1,mxndty
        do 1118 d=1,mxngr
        do 1115 b=1,mxdevt
        wprd(1,b,d,wi)=-1.0
1115    continue
1118    continue
1120    continue
        vpl=vpl+1
        ctsto='    '
```

Step 4: **The discovered reservoir data bank file is read. The production rate and the operating and maintenance costs are read.**

```
        do 1150 ii=1,18
1121    read(ifl,end=1122)    ipn,c1,rsty,ctst,f1,ctch,d,copt,imyr,
        (prd(i),i=1,imyr),(oam(i),i=1,imyr)
        go to 1123
1122    if(ifl.ne.24) call errmsg(4,601)
        ifl=25
        go to 1121
1123    if(ii.eq.1) then
```

Step 5: **GSAMID variables (‘ctst’ is play name, ‘rsty’ is the resource type, ‘ipn’ is the GSAM supply region and ‘f1’ is field size class for undiscovered reservoirs) from the bank files are stored in local variables.**

```
        ctsto=ctst
        rstyo=rsty
        ipno=ipn
        f1o=f1
        endif
```

Step 6: **Consistency checks are performed on the local variables.**

```
        if(ctsto.ne.ctst) call errmsg(4,491)
        if(ipno.ne.ipn) call errmsg(4,491)
        if(f1o.ne.f1) call errmsg(4,491)
        if(rstyo.ne.rsty) call errmsg(4,491)
        pname='                    '
        pname(1:4)=ctst
```

Step 7: **'po' and 'pname' (indicating play name) are stored in local variables.**

```
        if((po.ne.0).and.(pname.eq.pnameo)) then
        p=po
        else
        pnameo=pname
        do 1152 p=1,nply
        if(pname.eq.plynme(p)) go to 1153
1152    continue
        write(*,9152) pname
9152    format(' play: ',a20)
        call errmsg(4,453)
1153    continue
        endif
```

Step 8: **Consistency checks are performed on paygrade and resource type and error messages are printed.**

```
        if((d.le.0).or.(d.gt.3)) then
        write(*,1151) pname,f
1151    format(' Play: ',a20,' Field Size: ',i2)
        call errmsg(4,493)
        endif

        if((rsty.le.0).or.(rsty.gt.nrst)) then
        write(*,9951) v,vi,ipn,rsty,ctst,f1,ctch,d,copt,imyr,
        pname,nrst
9951    format(' v: ',2i8,a2,i3,a4,i3,a1,i3,a3,i3,a20,i5)
        call errmsg(4,494)
        endif
```

Step 9: **The variables 'wi' and 'b' are assigned values based on technology types (wi) and development type (b) respectively.**

```
        if(ctch.eq.'C') then
        wi=1
        elseif(ctch.eq.'M') then
        wi=2
        elseif(ctch.eq.'A') then
        wi=3
        elseif(ctch.eq.'E') then
        wi=3
        else
        write(*,1151) pname,f
        call errmsg(4,495)
        endif

        if(copt.eq.'P') then
        b=1
        elseif(copt.eq.'R') then
        b=2
        elseif(copt.eq.'I') then
        b=3
        else
        write(*,1151) pname,f
        call errmsg(4,496)
        endif
```

Step 10: **The values of 'wprd' and 'woam' are adjusted using 'prd' and 'oam' respectively. ‘wprd’ is in MMCF/year and ‘woam’ is in $MM/year.**

```
        if(wprd(1,b,d,wi).ne.-1.0) call errmsg(4,497)
        do 1140 i=1,mxnyr
        if(i.le.imyr) then
        wprd(i,b,d,wi)=prd(i)*1000.0
        woam(i,b,d,wi)=oam(i)*1000.0
        else
        wprd(i,b,d,wi)=0.0
        woam(i,b,d,wi)=0.0
        endif
1140    continue
1150    continue
        endif
```

Step 11: **Reservoirs for which production and O&M entries are not available in the reservoir bank file; ‘wprd’ and ‘woam’ variables are assigned as the primary case values.**

```
        if(vi.ne.0) then
        do 1365 wi=1,mxndty
        do 1360 d=1,mxngr
        do 1355 b=1,mxdevt
        if(wprd(1,b,d,wi).eq.-1.0) then
        if(wi.gt.1) then
        do 1351 i=1,mxnyr
        wprd(i,b,d,wi)=wprd(i,b,d,1)
        woam(i,b,d,wi)=woam(i,b,d,1)
1351    continue
        else
        call errmsg(4,606)
        endif
        endif
1355    continue
1360    continue
1365    continue
        endif
```

```
call gett(tmes(6),tmea(6),1)
```

Step 12: The subroutine ends.

```
return
end
```

Table of Contents

SUBROUTINE EXDVST

CALLED BY: EXPLPROD (Controls the main flow through the expl./dvlp. model.)

CALLS: PKNUDS (Packs the arrays containing the specifications of the development programs and saves the data no longer required to temporary files.)
DVLMSP (Estimates the MASP parameters for a selected reservoir.)
DRLCST (Estimates drilling cost for the play based on the drilling depth.)
PRJPAK (Packs the reservoir arrays.)
UDOPK (Packs/unpacks specification of secondary development options.)
INSU (Updates the pointers for the development programs.)
PRJSRT (Sorts the development reservoirs.)
GETT (Estimates the time required for each phase of the subroutine.)
ERRMSG (Prints out errors and warnings.)

CREATES: 'drill.out' (Outputs various drilling costs.)
'drilld.out' (Outputs various variables related to the MASP.)
'drille.out' (Outputs assorted variables: MASP related, expl. cost, etc.)

MAIN THEME: This routine computes the exploration and development decisions based on the supply price and various market constraints available. The routine performs the exploration and development decisions in three steps. First, it computes the drilling levels without any constraints. Second, it computes the actual drilling levels based on constraints such as footage constraints, technology penetration, etc. Third, it adjusts the parameters affected by the drilling decision and adjusts them for use in the next year.

Step 1: **'pknuds' opens the temporary file 'upsave.tmp' which would be used later for packing/unpacking data.**

```
        call pknuds(0,u)
```

Note: 'kflg(v)' is the flag for discovered reservoirs to indicate if further development of the field will never be economic in the future ('kflg' equal to zero means that the reservoir is economic).

‘nflg(v)’ is the flag for undiscovered (or discovered/undeveloped) reservoirs to indicate if the reservoir would be economic in the future (‘nnflg’ equal to zero indicates that the reservoir is economic). ‘uflg’ is the flag for secondary development to indicate if the reservoir would ever be economic in the future (‘uflg’ equal to zero means that the reservoir is economic). (Setting them to zero indicates that it may be possible that they are economic). ‘Nefl’ is total number of discovered reservoirs and undiscovered field size class and ‘kflg’ initializes the counter. ‘Nnfl’ is number of undiscovered field size class and ‘nflg’ initializes the counter. ‘Uflg’ initializes the counter for secondary development. ‘Nuds’ is total number of discovered reservoirs.

```
        do 15 v=1,nefl
        kflg(v)=0
15      continue

        do 20 g=1,nnfl
        nflg(g)=0
20      continue


        if (nuds.gt.0) then ! if discovered reservoirs then
        do 30 u=1,nuds
        do 25 wi=1,mxndtx
        uflg(wi,u)=0
25      continue
30      continue
        endif
```

Step 2: **The environmental counter is set to zero.**

```
        ienv=0
```

Note: The number of years is set.

```
        nyr=tme(ntme)-tmex+1
        if(nyr.le.0.or.nyr.gt.mxnyr) call errmsg(4,601)
```

Note: ‘vsldr’ is the total (fractional) decline in drilling costs for the entire time frame modeled.

```
        vsldr=drlvcs*(1.0-drlcim/100.0)**(nyr-1)/drlfcs
```

Note: All exploration and development drilling decisions are made in the '5000' loop on an annual basis. This loop encompasses the entire 'exdvst.for' routine.

```
        DO 5000 y=1,nyr    ! THE LARGE YEAR LOOP
```

Note: The following code sets the environmental regulations.

```
        if((tmex+y-1).ge.iyrenv(ienv+1).and.(tmex+y-1).lt.
        iyrenv(ienvcse+1))  ienv=ienv+1
9991    format(' Analyzing Year: ',i2,'  Environmental Code: ',i4)
```

Step 3: **The following cost factors are estima ted: cumulative decline in drilling costs (vdfc), the adjusted variable cost factor (accounts for the cost decline) (drlvcx), and the adjusted fixed cost factor (accounts for cost decline) (drlfcx).**

```
        vdfc=(1.0-drlcim/100.0)**(y-1)
        drlvcx=drlvcs*vdfc/drlfcs
        drlfcx=drlfcs*vdfc/drlfcs
        write(38,3899)y,drlfcx,drlvcx,vsldr,nyr,drlfcs,drlvcs
3899    format(1x,i5,1x,3(f8.5,2x),i5,2f8.5)
```

Step 4: **The 'do loop 150' is a play loop in which for the current year (y), the number of undiscovered reservoirs in a field size class (f), play (p), and year (y) is saved in a variable plysve (1,f,p,y). At the start of the run, the number of discovered undeveloped reservoirs (plysv(2,f,p,y)) and the number of reservoirs due to reserve growth (plysve(3,f,p,y)) are assigned to zero. All other variables (such as production, costs, drilling levels, etc.) for the current year (y) and play (p) are initialized.**

```
        do 150 p=1,nply
        sa=plyrga(p)
        xa=rst(p)

        do 140 f=1,mxnfsz
        plysve(1,f,p,y)=plynfl(f,p)
        plysve(2,f,p,y)=0.0
        plysve(3,f,p,y)=0.0
140     continue

        splprd(y,p)=0.0
        spldrl(y,p)=0.0
        spldcs(y,p)=0.0
        splndc(y,p)=0.0
        splwpr(y,p)=0.0
        spledr(y,p)=0.0
        sploec(y,p)=0.0
        spledc(y,p)=0.0
        splrsp(y,p)=0.0
        splrss(y,p)=0.0
        splirs(p)=0.0
        splenw(y,p)=0.0
        splnpw(y,p)=0.0
        spldwp(y,p)=0.0
        spldws(y,p)=0.0
        splrfc(y,p)=0.0
        splfom(y,p)=0.0
        splvom(y,p)=0.0
150     continue
```

Step 5: **The number of exploration/development decision (for a reservoir) is initialized (nprj=0) and the tax code is obtained for that year (this is 'h=taxcde(y)' which is currently modeled as the year when the royalty incentive is applicable.)**

```
        nprj=0
        h=taxcde(y)
```

Step 6: **Once various variables are initialized (in the earlier step) then the actual number of reservoirs are calculated from the discovered undeveloped category, or the reserves growth category based on the availability percentage. All the calculations are performed in the play loop '190' which is within the year loop 'DO 5000' as shown in step 2.**

```
        do 190 p=1,nply

        if (p.eq.58) then

1234    format(i4,f7.2,1x,13(f8.2,1x), i1)
        endif

        valo=0.0
```

Note: The supply region name (sa) and the resource type (xa) are obtained from the play counter (p). These entries come from the ply_dfn.spc file.

```
        sa=plyrga(p)
        xa=rst(p)
```

Note: Based on the availability percentage of the reservoirs under the reserve growth category, the reservoirs are assigned to the discovered undeveloped category (plybfl(f,p)). These reservoirs don't go through exploration economics.

```
        do 155 f=nfsz(p),1,-1


        If (plybfl_rg(f,p).gt.0.0) Then

        if (y.eq.1) then
        plybfl(f,p)=plybfl_rg(f,p)*availpen_rg(y,xa,sa)
        else
        plybfl(f,p)=plybfl(f,p)+plybfl_rg(f,p)*(availpen_rg(y,xa,sa)-
        availpen_rg(y-1,xa,sa))
        endif
        Endif

        valo=valo+plybfl(f,p)

        valo=valo*0.5

155     continue
```

Note: 'valo' is essentially that portion of 'valb' which is carried over to the next reservoir. 'valb' itself is that fractional portion of a reservoir that is carried over to the next reservoir. Since each reservoir is processed in quantum portions anything above or below that quantum is assigned to 'valb'. 'valb' itself has quantum limits and anything above or below those limits is assigned to 'valo'.

```
        valo=amin1(valo,0.0)
```

Note: At this stage the control is within the year loop (the 'DO 5000' year (y) loop), and the play loop (the 'DO 190' play (p) loop). Now the field size class loop starts ('DO 185' loop). The field size class loop starts from the biggest field size class (f=1 for field size class = 17, f=2 for field size class = 16 and so on) and ends in the smallest field size class (f=13 for field size class=5)

```
        do 185 f=1,nfsz(p)  !biggest to smallest field size class
```

Note: 'valo' is multiplied by two to reflect that the next bigger/smaller field size class is twice as large/small in size.

```
        valo=valo*2.0
        valos=valo
        valo=amin1(valo,0.5)
        valos=valos-valo
```

Note: If at least one reservoir is in the bank then it is processed. The error term is adjusted to reflect the larger field size. Here 'plybfl' is total number of 'nrrr' times reserve growth factor in 'resavrg.spc' and the 'undbnk' factor from the 'undbnk.spc' file. 'nrrr' is the number of undiscovered accumulations in an undiscovered field size class within a play; obtained from 'undisc.gsm' etc. files.

```
        if((plybfl(f,p)+valo).ge.0.5) then
```

Note: The total number of reservoirs in the field size class (f) and play (p) is assigned to the variable 'val'. In addition, the pointer to the corresponding undiscovered reservoir is assigned (g). Finally the pointer to the first possible reservoir (v) is assigned. 'v' starts from one to the total number undiscovered and discovered reservoir; i.e., 'v' is the absolute reservoir counter.

```
        val=plybfl(f,p)
        g=nnflp(f,p)
        v=nkpnt(g)
```

Note: The ratio (valr) of the number of reservoirs remaining (i.e., 'nnflpr – val')after assignment to the discovered undeveloped category (val) and the total number of undiscovered reservoirs is computed.

```
        if(nnflpo(f,p).gt.0.0) then
        valr=(nnflpr(f,p)-val)/nnflpo(f,p)
        else
        valr=0.0
        endif
```

Note: A fatal error message is printed if there are no reservoirs to process.

```
160     if(v.le.0) then
        call errmsg(4,711)
        endif
```

Note: The difference between the number of reservoirs not yet assigned for the specific type of reservoir and the expected number that should have been assigned at this point, is computed. If the difference rounds to at least one then it is assigned. The cumulative overflow in the equation is included and all assignments are made in integer steps. 'V' is a counter for GSAMID's starting from 1 (i.e., it is the absolute reservoir counter). 'Kdnwlb(v)' is number of undiscovered reservoirs in the field size class of the play, 'kdnwlc(v)' is original number of reservoirs left to be discovered and 'valb' is the number of reservoirs remaining to be developed including the residual for the play, 'kdnwla(d,v)' is the number of wells for primary development by pay grade, 'kdnwl(4,d,v)' is the number of wells (1-total, 2-already producing primary, 3-already producing secondary, 4-in reserve for discovered/undeveloped reservoirs converted to known).

```
valb=kdnwlb(v)-valr*kdnwlc(v)+valo
```

Note: Here if the value of 'valb' is greater than half the nearest integer value is calculated, this is reflected in 'ival'.

```
If (valb.ge.0.5) Then
ival=ifix(valb+0.5)
```

Note: The value of 'valo' reflects the difference in 'valb' and closest higher integer (ival).

```
valo=valb-ival
```

Note: 'valb' is reassigned the value of 'ival'.

```
valb=ival
```

Note: If this new integer value of 'valb' is greater than 'kdnwlb' the difference is stored in 'valo' and 'valb' is assigned the full value of 'kdnwlb'.

```
if(valb.gt.kdnwlb(v)) then

valo=valo+(valb-kdnwlb(v)) !actual excess

valb=kdnwlb(v)

endif
```

Note: Here the number of remaining undiscovered reservoirs after 'valb' gets discovered is calculated.

```
        kdnwlb(v)=kdnwlb(v)-valb
        nnflpr(f,p)=nnflpr(f,p)-valb
```

Note: The actual amount of discovery for a field size class and play is calculated and reflected in ‘val’.

```
        val=val-valb  !actual amt of discovery of f,p
```

Note: Here the actual number of wells drilled by pay grade and the absolute reservoir counter (v) is computed from the total wells that should be drilled in the pay grade for the full reservoir.

```
        do 165 d=1,mxngr

        kdnwla(d,v)=kdnwla(d,v)+kdnwl(4,d,v)*valb

165     continue

        Else

        valo=valb

        Endif
```

Note: Here the pointer to the next possible discovered reservoir that can be assigned is obtained and tested for a round-off error.

```
        v=kpnt(v)
```

Note: Here the number of reservoirs not fully assigned discovered reservoirs is saved and this process is continued to all assignments that are incomplete.

```
        plybfl(f,p)=val
```

Note: The control goes to statement 160 until ‘val’ (i.e., the remaining number of reservoirs for assignment) is less than one.

```
        if((v.gt.0).and.(val.ge.1.0)) go to 160
        endif
```

Note: The bank resources are accessed.

```
        g=nnflp(f,p)
        v=nkpnt(g)
```

Note: ‘Plysve(3,f,p,y)’ – OGIP of undiscovered reservoirs (1,) and OGIP of discovered/undeveloped reservoirs (2,) and OGIP of reserves growth reservoirs (3,) at beginning of year (BOY) for field size class, play, and year that is saved for reporting purposes in ‘Exdvso.for’.

```
        do 170 d=1,mxngr

        plysve(2,f,p,y)=plysve(2,f,p,y)+plybfl(f,p)
        *ndogip(d,g)
        plysve(3,f,p,y)=plysve(3,f,p,y)+plybfl_rg(f,p)
        *ndogip(d,g)*(1 - availpen_rg(y,xa,sa))
170     continue

        If (v.ne.0) Then
173     do 175 d=1,mxngr
```

Note: OGIP is updated by taking into account the number of wells drilled in the paygrade.

```
        if(kdnwl(4,d,v).gt.0) then
        plysve(2,f,p,y)=plysve(2,f,p,y) +
        kdnwla(d,v)*kdogip(d,v)/kdnwl(4,d,v)
        endif
175     continue

        v=kpnt(v)
        if(v.ne.0) go to 173
        Endif ! v

        valo=valo+valos
```

Note: The ‘185’ loop is the field size class loop.

```
185     continue
```

Note: The ‘190’ loop is the play loop.

```
190     continue  ! End Play Loop
```

Step 7: **At this stage the calculations are performed in the yearly loop of ‘DO 5000’ (the ‘y’ loop). Step 7 goes through unconstrained primary development indicated how much drilling could be**

done if there are no drilling footage constraints, no technology penetration constraints or other price constraints.

Note: Here the ‘gett’ routine determines the time spent in step 7.

```
        call gett(tmes(8),tmea(8),0)
```

Note: Here in the ‘210 do loop’ the control goes through all the discovered reservoirs (including discovered/undeveloped reservoirs) and identify the reservoirs where primary drilling could be done. If the MASP of such a reservoir is within bounds then it is added to the list of available reservoirs.

```
200     do 210 v=1,nefl
```

Note: The variables are initialized keeping track of the fraction of the reservoirs to be developed (kdo) and the fraction of reservoir to be developed by technology type (kdox) (i.e., current or advanced technology).

```
        kdo(v)=0.0
        do 201 wi=1,mxndtx
        kdox(wi,v)=0.0
201     continue
```

Note: The total number of wells that could be drilled using primary drilling methods (val) and the total number of possible primary wells (vala) for the reservoir is calculated. Processing is continued only if some primary development still required (i.e., val >0). 'kdnwlc(v)' is the original number of reservoirs to be discovered (this comes from the 'und*.gsm' file), 'kdnwla(d,v)' is the number of wells for primary development by pay grade, 'kdnwl(4,d,v)' is the number of wells (1-total, 2-already producing primary, 3-already producing secondary,4-in reserve for discovered/undeveloped reservoirs converted to known), 'kdnwlb(v)' is number of undiscovered reservoirs in the field size class of the play, 'kdnwlc(v)' is original number of reservoirs left to be discovered.

```
        val=0.0
        vala=0.0
        do 202 d=1,mxngr
        val=val+kdnwla(d,v)
        vala=vala+kdnwl(1,d,v)+
        float(kdnwl(4,d,v))*(kdnwlc(v)-kdnwlb(v))
202     continue

        if (val.gt.0.0) then
```

Note: The fraction of total wells to be developed in the reservoir is using the impact of rules indicating the minimum of wells per year and the maximum fraction per year (these are specified in the drl_cap.spc file). The maximum fraction of the wells (kdomx) that can be drilled is saved. 'Xdmnwl' is the minimum number of wells that could be drilled in a year for a reservoir and 'xddfrc' is the maximum fraction of total wells that can be drilled in a year

The logic for calculating ‘valb’: The maximum wells that could be developed for the field size class of the Play depending on the fraction (i.e. calculate xddfrc*vala) is determined. Then the maximum of either minimum well allowed to be drilled (xdmnwl or xddfrc*vala) is picked. Finally the minimum of this value for wells that can be drilled based on fraction constraints and those needing primary development (val) is picked.‘valb’ is the variable indicating the final number of variables drilled.

```
        valb=amin1(amax1(xdmnwl,(xddfrc*vala)),val)
        kdomx(v)=amax1(0.0,amin1(1.0,valb/val))
```

Note: Variables for play (p), region (s), and resource type (rsty), are obtained. If the tax regime has changed then the screening flags are reset.

```
        p=kdflpp(v)
        s=plyrga(p)
        rsty=kdrsty(v)
        ii4=kflg(v)/2**mxndtx
        if(ii4.ne.h) kflg(v)=h*(2**mxndtx)
```

Note: The ‘do loop 205’ loops through the technology options (in all cases it is 2; current and advanced), and gets the index to the specific technology (x), and checks to see if the specific technology has penetrated (dtcpen(y,x)). If it the average MASP parameters for primary development are computed.

```
        do 205 wi=1,mxndtx
        x=xdeqx(wi,rsty)
        if((x.ne.0).and.(dtcpen(y,x).gt.0.0)) then
```

Note: If the technology and penetration is positive we check to see if the reservoir has been screened out.

```
ii3=mod(kflg(v)/2**(wi-1),2)
if(ii3.ne.0) go to 205
```

Note: The MASP parameters are initialized and the code loops over all the pay grades (do loop 204) to estimate the weighted average parameters based on the number of wells in the pay grade and the present value of production from the wells.

```
tmasp(1)=0.0
tmasp(2)=0.0
tmasp(3)=0.0
val=0.0
vala=0.0
do 204 d=1,mxngr
if(kdnwla(d,v).gt.0.0) then
if(wi.eq.1) then
wj=1
else
wj=3
endif
```

Note: The 'dvlmsp' routine provides various entries for MASP, production etc.

```
call dvlmsp(1,0,v,y,wj,d,h,tmaspa,tmasp(4),vdfc,ienv)
```

Note: 'tmasp(1)' is the total dollars spent on variable drilling cost towards drilling in the pay grade.

```
tmasp(1)=tmasp(1)+tmaspa(1)*kdnwla(d,v)*tmasp(4)
```

Note: 'tmasp(2)' is the total dollar change per dollar change in drilling cost.

```
tmasp(2)=tmasp(2)+tmaspa(2)*kdnwla(d,v)*tmasp(4)
*kdnwla(d,v)
```

Note: 'tmasp(3)' is the total footage required to drill 'kdnwla' number of wells (including development dry holes).

```
        tmasp(3)=tmasp(3)+tmaspa(3)*kdnwla(d,v)
        val=val+kdnwla(d,v)*tmasp(4)
        vala=vala+kdnwla(d,v)
        endif
204     continue
        if(val.gt.0.0) then
```

Note: By dividing ‘tmasp (1)’ by ‘val’, ‘tmasp (1)’ is the MASP at variable drilling cost (xmasp).

```
        tmasp(1)=tmasp(1)/val
```

Note: By dividing ‘tmasp (2)’ by ‘vala’ and ‘val’, ‘tmasp (2)’ becomes the dollar change in MASP ($/MCF) per dollar change in drilling cost.

```
        tmasp(2)=tmasp(2)/vala/val
        endif
        xmasp=tmasp(1)
```

Note: The drilling costs for the reservoir(v) and the absolute MASP that can possibly occur in the future for the technology is calculated. It is determined if the reservoir ever be economic in future. If it is never going to be economic; it is removed from the list of prospective reservoirs. Cost of drilling at full costs (vddd) is set equal to ($/ft of drilling for the reservoir/full cost factor (from the ‘drl_cap’.spc file; generally one) * depth required from all wells)

```
        call drlcst(1,kddpth(v),drlfcy,s)
        vddd=drlfcy/drlfcs*tmasp(3)
        if(vddd*tmasp(2).lt.0.0)vddd=0.0
```

Note: ‘xmaspx’ is the minimum MASP that the reservoir would ever be at during the time frame modeled.

```
        xmaspx=((tmasp(1)-(drlvcx*vddd*tmasp(2)))*
        amin1(1.0,(dtccsf(nyr,x)/dtccsf(y,x)))+
        (vsldr*vddd*tmasp(2)))
```

Note: Here the code checks to see if the minimum MASP (xmaspx) is going to be higher than the well head price (supprc(yx,s)), if it is, then that reservoir is skipped entirely.

```
        do 206 yx=y,nyr
        if(xmaspx.le.supprc(yx,s)) go to 207
206     continue
        kflg(v)=kflg(v)+2**(wi-1)

207     continue
```

Note: If the long term masp (i.e., xmaspx)is less than the well head price then , the current masp at lowest (variable) drilling costs versus current supply price is checked in the current year.

```
        if(xmasp.le.supprc(y,s)) then

        if(nprj.ge.mxnprj) then
        call prjpak
        endif
        nprj=nprj+1
        if(nprj.gt.mxnprj) call errmsg(4,602)
        bpnt(nprj)=v
        breg(nprj)=s
        if(tmasp(2).le.0.0) then
        write(*,9995) v,x,(tmasp(ix),ix=1,3),val,xmaspx
9995    format(' tmasp - v,x: ',i9,i3,5(1x,f10.3))
        call errmsg(3,951)
        tmasp(2)=0.0001
        endif
```

Note: If the MASP is less than the wellhead price development is carried over in the year. 'Bsrfl(nprj)' is set to breakeven drilling cost, the highest drilling cost where the reservoir is still economic. ‘bsrfl(nprj)’ is the extra dollars (above variable cost) that can be spent on drilling cost and still break even for the reservoir. If the MASP is less than the wellhead price, the reservoir parameters are stored so that sorting can be done at a later step (step 12).

```
        bsrfl(nprj)=(supprc(y,s)-xmasp)/tmasp(2)
        -drlvcx
        bdelt(nprj)=tmasp(2)
        btec(nprj)=wi
        bdrl(nprj)=tmasp(3)
        bnfl(nprj)=1.0
        bs(nprj)=nprj
        endif
        endif
205     continue
        endif

210     continue
        call errmsg(1,921))
```

Note: Only the year loop (DO 5000) is active at this stage indicating all secondary development decisions to be made on a yearly basis.

Step 8: **Step 8 goes through evaluation of infill/recomplete opportunities for previously developed reservoirs (i.e., for already producing reservoirs).**

```
        if(nuds.gt.0) then
        u=upfrst
```

Note: Here the code loops through the reservoirs and only looks those reservoirs where secondary development is incomplete (i.e., the year for secondary development, upsyr(u) = 0).

```
300     if(upsyr(u).eq.0) then
```

Note: Here the array switch that keeps track of secondary development is initialized.

```
        do 303 b=1,mxdevt
        do 302 d=1,mxngr
        udo(b,d,u)=0.0
        do 301 wi=1,mxndtx
        udox(d,wi,u)=0.0
301     continue
302     continue
303     continue
```

Note: The year (y1) that primary development started, index of the technology (x1) used for primary development, and the technology resource combination for that technology (wj) e.g., is obtained. If the primary technology is advanced (wj=2) then the secondary must be advanced and vice versa.

```
        y1=upyr(u)-tmex+1
        x1=uptchp(u)

        if(upcde(u).gt.nefl) then
        g=upcde(u)-nefl
        rsty=ndrsty(g)
        else
        v=upcde(u)
        rsty=kdrsty(v)
        endif
        wj=xdeqv(x1,rsty)+1
        if(mxndtx.ne.2) call errmsg(4,701)
        if(wj.eq.2) then
        wj=1
        elseif(wj.ne.1) then
        write(*,9961) u,y1,uptchp(u),rsty
9961    format(' u,y1,uptchp,rsty: ',4i6)
        call errmsg(4,702)
        endif
```

Note: The variables on index to reservoir description, play, size category, reservoir type, depth and the window year for secondary development are assigned. In the exploration and production runs, ‘upcde’ will always be less than or equal to ‘nefl’ and hence the ‘else’ piece of the following ‘if’ statement would be active.

```
        if(upcde(u).gt.nefl) then
        v=0
        g=upcde(u)-nefl
        p=nnflb(g)/mxnfsz
        f=nnflb(g)-p*mxnfsz+1
        rsty=ndrsty(g)
        ic=3
        i=g
        dpth=nddpth(g)
        y2=y1+ndwin(wj,g)
        y3=y2
        else
        v=upcde(u)
        g=0
        f=0
        p=kdflpp(v)
        rsty=kdrsty(v)
        ic=1
        i=v
        dpth=kddpth(v)
        y2=y1+kdwin(wj,v)
        y3=y2
        endif
        s=plyrga(p)
```

Note: The code loops over pay grades (d), technologies (wi) secondary development options to determine what is available and economic.

```
        do 307 d=1,mxngr
```

Note: It is ensured that the current year (y) is within the development window (i.e., between y2 and y3) and if it is not then it is ignored. Currently in the exploration and production module, y2 and y3 are the same and are the year when secondary development occurs. It is also ensured that wells exist in the pay grade for the development program.

```
if((y2.le.y).and.(y3.ge.y)) then
if(upnwl(d,u).gt.0.0) then
```

Note: The code loops over technology combinations. 'X' set to the index of the technology to be used for secondary development. 'Wj' is set to the technology option (Current, Advanced) that the technology represents. It is ensured that the technology exists (i.e., 'x'>0), that the index of the primary technology for the technology combination is the same as that used in the actual primary development, and that the technology to be tested for secondary development has penetrated. The option is ignored if it is not.

```
do 306 wi=1,mxndty
x=xdeqy(2,wi,rsty)
wj=xdeqv(x,rsty)+1
if((x.ne.0).and.(xdeqy(1,wi,rsty).eq.x1).and.
(dtcpen(y,x).gt.0.0)) then
```

Note: The screening flag is checked. It is reset if it set under a different tax regime. Secondary development options for development program are ignored if the flag is set.

```
ii4=uflg(wj,u)/2**mxngr
if(ii4.ne.h) uflg(wj,u)=h*(2**mxngr)
ii4=mod(uflg(wj,u)/2**(d-1),2)
if(ii4.eq.1) go to 306
```

Note: The ‘dvlmsp’ outline provides the MASP components.

```
call dvlmsp(ic,1,i,y,wi,d,h,tmaspb,tmspb4,vdfc,ienv)
ii3=1
```

Note: The drilling cost (‘drlfcy’ $/ft) for the reservoir is obtained and the code loops over secondary development options.

```
call drlcst(1,dpth,drlfcy,s)

do 305 b=mxdevt,1,-1
```

Note: It is estimated if secondary development will ever be economic. If so then the flag (ii3) is set to zero.

```
      if(tmaspb(3,b).gt.0.0)then
      vddd=drlfcy/drlfcs*tmaspb(3,b)
      else
      vddd=.0001
      endif
      if(ii3.ne.0) then
```

Note: The minimum achievable MASP for secondary development is calculated.

```
      xmaspx=((tmaspb(1,b)-(drlvcx*vddd*tmaspb(2,b)))*
      amin1(1.0,(dtccsf(nyr,x)/dtccsf(y,x)))+
      (vsldr*vddd*tmaspb(2,b)))
```

Note: The lowest possible masp (xmaspx) is checked against the wellhead price.

```
      do 304 yx=y,nyr
      if(xmaspx.le.supprc(yx,s)) ii3=0
304   continue
      endif
```

Note: The pay grade/technology option flag is set and the MASP is obtained. If MASP is less than the supply price then secondary development could be carried over and is stored for later sorting and processing

```
      bd=d*3+b
      xmasp=tmaspb(1,b)


      if(xmasp.le.supprc(y,s)) then
      if(nprj.ge.mxnprj) then
      call prjpak
      endif
      nprj=nprj+1
      if(nprj.gt.mxnprj) call errmsg(4,602)
      bpnt(nprj)=mxnefl+u
      breg(nprj)=s
```

Note: The breakeven drilling cost is set. If there is no (i.e., for refrac only) then the breakeven drilling cost is set to infinity.

```
         if(tmaspb(2,b).gt.0.0) then

         bsrfl(nprj)=(supprc(y,s)-xmasp)/tmaspb(2,b)
         -drlvcx

         else

         bsrfl(nprj)=9999999.9
         tmaspb(2,b)=0.0001

         endif

         bdelt(nprj)=tmaspb(2,b)
         btec(nprj)=wj+bd*(mxndty+1)
         bdrl(nprj)=tmaspb(3,b)*upnwl(d,u)
         bnfl(nprj)=1.0
         bs(nprj)=nprj
         endif
305      continue
```

Note: The screening flag is set.

```
         if(ii3.eq.1) then
         uflg(wj,u)=uflg(wj,u)+2**(d-1)
         endif
         endif
306      continue
         endif
         endif
307      continue
         endif
```

Note: Here the code goes on to the next discovered reservoir to check secondary development options.

```
         u=upf(u)
         if(u.gt.0) go to 300
         endif
         call errmsg(1,922)
```

Note: Here the code goes through the list of all undiscovered and discovered/undeveloped reservoir and estimates the MASP to be used for exploration. Since all discovered/undeveloped reservoirs are discovered prospects, the code that generates development reservoirs for the discovered/undeveloped reservoirs is skipped.

```
        do 450 g=1,nnfl
```

Note: The share of discovered/undeveloped reservoirs to be developed in a year is initialized. ‘gdo(g)’ is specific to the reservoir counter and ‘gdox(wi,g)’ is specific to technology type and reservoir counter.

```
        gdo(g)=0.0
        do 401 wi=1,mxndtx
        gdox(wi,g)=0.0
401     continue
```

Note: The play (p), region (s), field size class (f) and resource type (gsty) are obtained and it is checked if any discovered/undeveloped or undiscovered reservoirs still exist for the play/size class.

```
        p=nnflb(g)/mxnfsz
        s=plyrga(p)
        f=nnflb(g)-p*mxnfsz+1
        rsty=ndrsty(g)
        if((plybfl(f,p)+plynfl(f,p)).gt.0.0) then
```

Note: The tax regime that the screening flag is estimated under is obtained and reset if the tax regime has changed.

```
        ii4=nflg(g)/2**mxndtc
        if(ii4.ne.h) nflg(g)=h*(2**mxndtc)
```

Note: The code loops over primary technology options for a reservoir, the technology type is obtained and it is ensured that the technology penetration is positive. If the technology penetration is positive, the screening flag is checked and if it has not been set, the weighted average MASP for the reservoir based on the number of wells by pay grade and PV of production from these wells is estimated.

```
        do 410 wi=1,mxndtx
```

Note: The technology type is obtained and technology penetration is checked.

```
        x=xdeqx(wi,rsty)
        if((x.ne.0).and.(dtcpen(y,x).gt.0.0)) then
```

Note: The screening flag is checked.

```
        ii3=mod(nflg(g)/2**(x-1),2)
        if(ii3.ne.0) go to 410
```

Note: The MASP arrays and variables are initialized.

```
        ndmasp(1,wi,g)=0.0
        ndmasp(2,wi,g)=0.0
        ndmasp(3,wi,g)=0.0
        ndmasp(4,wi,g)=0.0
        val=0.0
        vala=0.0
```

Note: The values for the weighted average MASP calculation are accumulated.

```
        do 404 d=1,mxngr
        if(ndnwl(d,g).gt.0.0) then
        if(wi.eq.1) then
        wj=1
        else
        wj=3
        endif

        call dvlmsp(3,0,g,y,wj,d,h,tmasp,tmasp(4),vdfc,ienv)
        ndmasp(1,wi,g)=ndmasp(1,wi,g)+tmasp(1)*ndnwl(d,g)*tmasp(4)
        ndmasp(2,wi,g)=ndmasp(2,wi,g)+tmasp(2)*ndnwl(d,g)*tmasp(4)
        *ndnwl(d,g)
        ndmasp(3,wi,g)=ndmasp(3,wi,g)+tmasp(3)*ndnwl(d,g)
        ndmasp(4,wi,g)=ndmasp(4,wi,g)+tmasp(4)*ndnwl(d,g)
        val=val+ndnwl(d,g)*tmasp(4)
        vala=vala+ndnwl(d,g)
        endif

404     continue
        if(val.le.0) call errmsg(4,953)
```

Note: The maximum wells drilled in a year is set based on development drilling parameters.

'valb' is the maximum number of wells in one reservoir.

```
        valcd=vala
        valb=amin1(amax1(xdmnwl,(xddfrc*vala)),vala)
        if((plybfl(f,p).lt.1.0).and.(plybfl(f,p).gt.0.0)) then
        vala=vala*plybfl(f,p)
        endif

        if(vala.gt.0.0) then
        gdomx(g)=amax1(0.0,amin1(1.0,valb/vala))
        else
        gdomx(g)=1.0
        endif
        gdomx(g)=gdomx(g)*plybfl(f,p)
```

Note: The weighted average MASP parameters is estimated.

```
        ndmasp(1,wi,g)=ndmasp(1,wi,g)/val
        ndmasp(2,wi,g)=ndmasp(2,wi,g)/valcd/val
```

Note: The drilling cost for the reservoir are obtained and the screening flag is reset. The masp is adjusted for full change in non-drilling costs (dtccsf) and for the full change in variable drilling costs in the last year of analysis.

```
        call drlcst(1,nddpth(g),drlfcy,s)
        if(drlfcs.eq.0.0) call errmsg(3,811)
        vddd=drlfcy/drlfcs*ndmasp(3,wi,g)
        if(vddd*ndmasp(2,wi,g).lt.0.0)then
        call errmsg(3,812)
        vddd=0.1
        endif
        if(dtccsf(y,x).eq.0.0) call errmsg(3,801)
```

Note: The minimum achievable MASP is calculated.

```
        xmaspx=((ndmasp(1,wi,g)-(drlvcx*vddd*ndmasp(2,wi,g)))*
        amin1(1.0,(dtccsf(nyr,x)/dtccsf(y,x)))+
        (vsldr*vddd*ndmasp(2,wi,g)))
```

Note: The lowest possible masp (xmaspx) is checked against the wellhead price.

```
        do 406 yx=y,nyr
        if(xmaspx.le.supprc(yx,s)) go to 407
406     continue
        nflg(g)=nflg(g)+2**(x-1)

407     continue
        endif
410     continue
```

```
endif
```

Note: The ‘DO 5000’ year loop is the only active loop at the start of Step 10.

Step 10: **This step determines the exploration levels.**

Note: The code goes through all the exploration prospects, estimates the long term expected value of the exploration prospect, and if there is some chance that the prospect will be economic it includes it in the list of available prospects to be done.

Note: The code loops through all the plays.

```
do 550 p=1,nply
```

Note: The resource types (rsty) and GSAM supply region (s) are assigned.

```
rsty = rst(p)
s=plyrga(p)
```

Note: Various counters are assigned.

```
g1 = nnflp(nfsz(p),p)

iv1=nkpnt(g1)

do 502 z=1,netc
if(etcpen(y,z).gt.0.0) then
```

Note: The exploration drilling cost at full initial drilling rates is set.

```
call drlcst(2,etcdep(p),drlfcy,s)
```

Note: ‘vddd’ and ‘etcdrl(z,p)’ is the total dollars needed (full cost) for drilling (specific to the play and technology) for exploration.

```
vddd=drlfcy*etcdep(p)/drlfcs
```

```
etcdrl(z,p) = drlfcy*etcdep(p)/drlfcs
```

Note: The total exploration cost (full cost) is saved in the ‘etcdrl1’ variable. In addition, the non-drilling portion of the exploration cost as indicated in exp_cst.spc file (etccst) is stored in the ‘etccst1’ variable.

```
etcdrl1=etcdrl(z,p)
etccst1=etccst(z,p)
```

Note: The variable exploration drilling costs is calculated and saved in the etcdrl(z,p) variable.

```
etcdrl(z,p) = etcdrl(z,p)*drlvcx

if (ienv.gt.0.and.iv1.ge.1) then
```

Note: Costs due to environmental regulations are added to the exploration drilling cost.

```
etcdrl(z,p)=etcdrl(z,p)+envdcnc(ienv,iv1)+
envdcf(ienv,iv1)*etcdep(p)
etcdrl1=etcdrl1+envdcnc(ienv,iv1)+
envdcf(ienv,iv1)*etcdep(p)
```

Note: The environmental non-drilling costs are added to the etccst1 variable.

```
etccst1=etccst1+envndnc(ienv,iv1)
endif
```

Note: The total exploration cost (exmasp(1,z,p)) is obtained by adding both drilling and non-drilling exploration costs. This is done for the variable cost.

```
exmasp(1,z,p)=etccst1+etcdrl(z,p)
```

Note: ‘exmasp(2,z,p)’ is the increase in NPV in drilling cost with dollar increase in drilling cost..

```
        exmasp(2,z,p)=(etcdrl1-etcdrl(z,p))*ndpvds(1,2,1,iv1)
```

Note: ‘exmasp(3,z,p)’ is the total amount of successful drilling footage needed inclusive of drilling efficiency. The drilling efficiency reduces the footage required to drill for exploration since it is assumed that exploration drilling is conducted faster than development drilling.

```
        exmasp(3,z,p)=etcdep(p)/drleff
        endif
502     continue
```

Note: The region is obtained and the code loops through size classes for the play representing the number of exploration vectors available.

```
        if(nfsz(p).gt.0) then
        do 535 f=1,nfsz(p)
```

Note: The amount of exploration selected for this exploration step is initialized. It is ensured that the undiscovered resource exists in the field size class and if not calculations proceed to the next field size class.

```
        g=nnflp(f,p)
        fpdo(g)=0.0
        if((g.le.nnfl).and.(plynfl(f,p).gt.0.0)) then
```

Note: The code loops through the exploration technologies available and initializes the amount of exploration (fpdoz(z,g,)) selected for this step and continues if the exploration technology penetration is positive.

```
        do 530 z=1,netc
        fpdoz(z,g)=0.0

        if (etcpen(y,z).gt.0.0) then
```

Step 10h: **The exploration vector (as specified in ‘ply_dfn.spc’ file) is re-specified from specification by field to specification by reservoir and normalized so that for each successful exploration finds one package containing three reservoirs. 1 is the biggest field size class and 2 the smaller field size class.**

```
        if ((etcflf(f,f,z).gt.0.0).and.
        (etcrst(z).eq.plyrst(p))) then
        vlflft=0.0
        vlfo=0.0
        valmx=0.0
        do 504 f1=1,nfsz(p)
        vlfn=amax1(0.0,plynfl(f1,p)-vlfo)
        vlfo=vlfn*2.0
        if(f1.ge.f) then
        if(f1.eq.f) valmx=vlfn
        vlflf(f1)=etcflf(f1,f,z)*amax1(vlfn,0.001)
        if(etcflfp(f1,f1,z,p).gt.0.0) vlflf(f1)=
        etcflfp(f1,f,z,p)*amax1(vlfn,0.001)
        else
        vlflf(f1)=0.0
        endif
504     continue

        do 507 f1=nfsz(p),f,-1
        if(f1.gt.f) then
        vlflf(f1)=vlflf(f1)+2.0*vlflf(f1-1)
        endif
        vlflft=vlflft+vlflf(f1)
507     continue

        if(vlflf(f).le.0.0) call errmsg(3,862)
        if(vlflft.le.0.0) call errmsg(3,861)
        vlflft=(1.0+2.0)/vlflft

        do 506 f1=f,nfsz(p)
        vlflf(f1)=vlflf(f1)*vlflft
506     continue
```

Note: The maximum number of successful wells available for the exploration step is set and the weighted average MASP of the exploration step including the cost of exploration with the expected MASPs of the average number of fields to be found with exploration is computed. The best technology with the maximum penetration at that technology is chosen which is then followed by the next best technology.

```
        valmx=valmx/vlflf(f)
```

Note: The variables used to compute MASP are initialized.

```
        vlmasp=0.0
        vldelt=0.0
        vldrl=0.0
        vltot=0.0
        vlprd=0.0
        yfld=0.0
```

Note: The code loops over all the field size classes found with the exploration step.

```
do 525 f1=f,nfsz(p)
```

Note: The ‘number of reservoirs’ in the size class found with the exploration step is obtained and if positive then the code continues.

```
xfld=amin1(plynfl(f1,p),valmx*vlflf(f1))
yfld=yfld+xfld

if(xfld.gt.0.0) then
g1=nnflp(f1,p)
```

Note: The code goes through the technologies available for the resource type with positive penetration and estimates the MASP at full drilling cost. The selection is made to determine which technology is better. 'Xm' is the (adjusted) development masp at variable drilling cost. If the development technology penetration has not occurred, the development MASP is made higher than the well head price by $1/MCF; and hence is made unavailable in the current year.

```
do 510 wi=1,mxndtx
x=xdeqx(wi,rsty)
if((x.ne.0).and.(dtcpen(y,x).gt.0.0).and.
(g1.le.nnfl)) then
call drlcst(1,nddpth(g1),drlfcy,s)
vddd=drlfcy/drlfcs*ndmasp(3,wi,g1)

xm(wi)=ndmasp(1,wi,g1)+ndmasp(2,wi,g1)
*(drlfcx-drlvcx)*vddd
else
xm(wi)=supprc(y,s)+1.0
endif

510    continue
valmn=1.0
```

Note: The code goes through the various technology options and finds the technology with the smallest MASP.

```
do 520 x1=1,mxndtx
wi=1
do 515 x2=1,mxndtx
if((x2.eq.1).or.(xm(x2).lt.xm(wi))) then
```

```
        wi=x2
        endif
515     continue
```

Note: If MASP for the technology type selected for the reservoir is less than supply price, then values are accumulated and the price set so that it cannot be chosen again.

```
        if(xm(wi).le.supprc(y,s)) then
        x=xdeqx(wi,rsty)
```

Note: Here the code accounts for the Federal Lands Technology Penetration Curve. 'Dtcpenyx' is the effective development penetration rate.

```
        dtcpenyx =  dtcpen(y,x)*(1-frac_fed(iv1)) +
        (fedpend(y,x)+dtcpen(y,x))*frac_fed(iv1)
```

Note: The code assigns the exploration step based on the share remaining (valmn) and the development technology penetration rate.

```
        val=amin1(valmn,dtcpenyx)
```

Note: The weighted average development MASP is computed.

```
        vlmasp=vlmasp+xm(wi)*xfld*val*ndmasp(4,wi,g1)
        vldrl=vldrl+ndmasp(3,wi,g1)*xfld*val
        vldelt=vldelt+ndmasp(2,wi,g1)
        *xfld*val*ndmasp(4,wi,g1)
        *ndmasp(3,wi,g1)*xfld*val
        vlprd=vlprd+ndmasp(4,wi,g1)*xfld*val
        vltot=vltot+xfld*val
        valmn=valmn-val
```

Note: The option is marked so that it is not selected again on this pass.

```
        xm(wi)=supprc(y,s)+1.0
        endif
520     continue
        endif
525     continue  ! Field Size Class Loop
```

Note: The exploration MASP (vlmasp) is computed for both the technologies by taking into account the success rate of exploration (etcsrt(f,z,p)) and the cost of exploration (exmasp(2,z,p)).

```
if((vltot.gt.0.0).and.(vlprd.gt.0.0)) then
if(valmx.le.0.) call errmsg(3,888)
if(etcsrt(f,z,p).le.0.) call errmsg(3,889)
if(vldrl.le.0.) call errmsg(4,988)
vlmasp=(exmasp(1,z,p)/etcsrt(f,z,p)*yfld+vlmasp)
/vlprd
```

Note: ‘vldelt’ is the total exploration cost for the play (it changes by development technology) above variable cost.

```
 vldelt=(exmasp(2,z,p)/etcsrt(f,z,p)*yfld
+vldelt)/vlprd/vldrl

12345    format(1x,4i5,2f12.4,i8,f12.5)
```

Note: ‘vldrl’ is total footage needed for exploration drilling by taking into account the exploration success rate.

```
vldrl=exmasp(3,z,p)/etcsrt(f,z,p)
else
vlmasp=supprc(y,s)+1.0
endif
```

Note: If the long term masp (i.e., vlmasp)is less than the well head price then , the current masp at lowest (variable) drilling costs versus current supply price is checked in the current year.

```
if(vlmasp.le.supprc(y,s)) then
if(nprj.ge.mxnprj) then
call prjpak
endif
nprj=nprj+1
if(nprj.gt.mxnprj) call errmsg(4,602)
bpnt(nprj)=mxnefl+mxnuds+mxnnfl+g
breg(nprj)=s
btec(nprj)=z
bnfl(nprj)=valmx
bs(nprj)=nprj
if(vldelt.le.0.0) call errmsg(3,997)
```

Note: If the MASP is less than the wellhead price exploration is carried over in the year. 'Bsrfl(nprj)' is set to the breakeven drilling cost, the highest drilling cost where the reservoir is still economic. ‘bsrfl(nprj)’ is the extra dollars (above variable cost) that can be spent on drilling cost and still break even for the reservoir. If the MASP is less than the wellhead price, the reservoir parameters are stored so that sorting can be done at a later step (step 12).

```
        bsrfl(nprj)=(supprc(y,s)-vlmasp)/vldelt
        -drlvcx
        bdelt(nprj)=vldelt
        bdrl(nprj)=vldrl
        endif
        endif
        endif
530     continue
        endif
535     continue
        endif
550     continue
        call errmsg(1,924)
```

Step 11: **Regional drilling footage capacities are assigned to ‘drtdrg’ and ‘drterg’ variables based on development and exploration footage capacity specified in ‘drl_rcp.spc’ file for the first year. From the second year onwards the footage capacities available are the maximum of last year’s capacity (drldrg(s,y-1)) or 20% of last year’s total capacity available.**

```
        drldnt(y)=0.0
        drlent(y)=0.0
        drtdnt(y)=0.0
        drtent(y)=0.0
        do 610 s=1,nsrg
        drldrg(s,y)=0.0
        drlerg(s,y)=0.0
        if(y.eq.1) then
        drtdrg(s,y)=drlrcp(s,2)
        drterg(s,y)=drlrcp(s,1)
        else
        drtdrg(s,y)=amax1(drldrg(s,y-1),drtdrg(s,y-1)*drlchf/100.0)
        drterg(s,y)=amax1(drlerg(s,y-1),drterg(s,y-1)*drlchf/100.0)
        endif
        drtdnt(y)=drtdnt(y)+drtdrg(s,y)
        drtent(y)=drtent(y)+drterg(s,y)
        drcdrg(y,s)=drlvcx
        drcerg(y,s)=drlvcx
610     continue
        call errmsg(1,925)
```

Step 12: **All the drilling reservoirs are sorted in descending order of drilling cost (above variable cost) where the reservoir is still economic.**

```
        call prjsrt
        call errmsg(1,926)
        call gett(tmes(8),tmea(8),1)
```

Step 13: **Here the code goes through reservoirs in descending order of drilling cost for all reservoirs that are still economic and computes actual drilling levels based on constraints available.**

Note: The regional drilling footage levels and the share of reservoirs that will be implemented in the year based on the available drilling capacity are determined. This section is run twice, first to look at economics where drilling capacity is not allowed to move between regions (jfl=0) and then when it is (jfl=1).

```
        call gett(tmes(9),tmea(9),0)
1000    if(nprj.gt.0.0) then
```

Note: The flag indicating the pass through the assignment loop is initialized and the following key fractional variables are set such as: drilling after retirement (xrrt), when drilling cost go below full cost (xchf) minimum drilling that must stay in the region (xchv), maximum increase in national drilling capacity (xchx), maximum increase in regional drilling capacity (xchy), total (national) drilling footage available for reassignment (drxnt) and amount of drilling reassigned (drunt) because of rig movement.

```
        jfl=0
        xrrt=(1.0-drlrrt/100.0)
        xchf=xrrt*drlchf/100.0
        xchv=xrrt*drlchv/100.0
        xchx=drlchx/100.0
        drxnt=(drtdnt(y)+drtent(y))*(xchx-xrrt)
        drunt=0.0
```

Note: Here the code goes through development/exploration reservoirs in sorted order.

```
1010    continue

        do 1090 a1=1,nprj
        a=bs(a1)! back pointer
```

Note: The specifications of selected reservoirs are obtained, for example, the type of development (b), pay grade (d) and the number of wells (xfld).

```
        s=breg(a)
```

```
bd=btec(a)/(mxndty+1)
wi=btec(a)-bd*(mxndty+1)
d=(bd-1)/3
b=bd-d*3
xfld=bnfl(a)
xdrlmn=0.0
if(xfld.ge.0.00001) then
```

Note: The pointers to discovered reservoirs, undiscovered/banked reservoirs, development programs (for secondary development), or exploration reservoirs are obtained as appropriate.

```
ii=bpnt(a)
v=0
u=0
g=0
f=0
p=0
iexpf=0
vmin=drcdrg(y,s)

if(ii.le.mxnefl) then!For development drilling
v=ii
rsty=kdrsty(v)
dpth=kddpth(v)
p=kdflpp(v)
elseif(ii.le.(mxnefl+mxnuds)) then  !For Infill
u=ii-mxnefl
if(upcde(u).le.nefl) then
rsty=kdrsty(upcde(u))
dpth=kddpth(upcde(u))
p=kdflpp(upcde(u))
else
rsty=ndrsty(upcde(u)-nefl)
dpth=nddpth(upcde(u)-nefl)
p=kdflpp(upcde(u)-nefl)
endif
elseif(ii.le.(mxnefl+mxnuds+mxnnfl)) then   !Disc/Undev.
g=ii-mxnefl-mxnuds
rsty=ndrsty(g)
dpth=nddpth(g)
p=nnflb(g)/mxnfsz
f=nnflb(g)-p*mxnfsz+1
elseif(ii.le.(mxnefl+mxnuds+2*mxnnfl)) then  !exploration
g=ii-mxnefl-mxnuds-mxnnfl
rsty=ndrsty(g)
dpth=nddpth(g)
p=nnflb(g)/mxnfsz
f=nnflb(g)-p*mxnfsz+1
if(g.gt.nnfl) call errmsg(3,995)
vmin=drcerg(y,s)
iexpf=1
else
call errmsg(4,603)
endif
```

Note: The drilling costs and technology specifications are obtained.

```
yfld=0.0
if(iexpf.eq.0) then

x=xdeqx(wi,rsty)
xa=x
if(xa.gt.mxrsty)xa=xa-mxrsty
if(xa.lt.1.or.xa.gt.mxrsty) call errmsg(4,693)

else

x=btec(a)
xa=x
if(xa.gt.mxrsty)xa=xa-mxrsty
if(xa.lt.1.or.xa.gt.mxrsty)call errmsg(4,693)

b=0
d=0
endif
z=x
```

Note: It is ensured that the reservoir is economic at regional drilling cost.

```
xbsrfl=bsrfl(a)+drlvcx
drltcs = dr_rig(s)
if(xbsrfl.ge.vmin) then
```

Note: The drilling capacity actually used (val1) and drilling capacity available (val2) are obtained.

```
if(iexpf.eq.0) then
val1=drldrg(s,y)
val2=drtdrg(s,y)
else
val1=drlerg(s,y)
val2=drterg(s,y)
endif
```

Note: The amount of drilling capacity available at the breakeven drilling cost for the reservoir is obtained. 'xdrl' is set to capacity in the region and 'ydrl' is set to capacity transported from other regions. ‘yrdl’ is assigned to zero since here evaluations are done without any regional rig movement.

```
if(xbsrfl.le.drlvcx) then
```

Note: The breakeven cost less than variable cost is set.

```
xdrl=val1
ydrl=0.0

elseif(xbsrfl.lt.drlfcx) then
```

Note: The breakeven cost between the variable and full cost is calculated by finding the capacity available at that cost. This is done by interpolating between the capacity at the variable cost and the minimum capacity at full cost using variable, full and breakeven drilling costs.

```
xdrl=(xbsrfl-drlvcx)/(drlfcx-drlvcx)*
(val2*(xchf-xchv))+val2*xchv
ydrl=0.0
elseif(xbsrfl.lt.(drlfcx+drltcs)) then
```

Note: The breakeven cost between full price and price needed to transport capacity to the region is calculated. The available capacity is set assuming maximum builds in the region and a proportional fraction of transport of capacity to the region from elsewhere.

```
xdrl=(val2*xchx)
ydrl=amax1(0.0,((xbsrfl-drlfcx)/drltcs)*drxnt-drunt)

else
```

Note: The breakeven cost above full cost and transport cost is calculated assuming all the capacity can move.

```
xdrl=(val2*xchx)
ydrl=amax1(0.0,(drxnt-drunt))

endif
```

Note: The transport capacity that can be moved to the region based on regional growth fractions is constrained and the capacity increases are combined into one variable 'xdrl'.

```
ydrl=amax1(0.0,amin1(ydrl,(val2*dr_reg(s)-amax1(xdrl,val1))))
xdrl=amax1(0.0,(xdrl-val1))+ydrl
```

Note: The maximum fraction of a reservoir that can be implemented based on drilling capacity is determined.

```
if(bdrl(a).le.0.0) then
vscl=1.0
else
```

Note: 'Vscl' is calculated at the breakeven drilling cost.

```
vscl=amin1(1.0,amax1(0.0,(xdrl/(xfld*bdrl(a)))))
endif
```

Note: Development is determined based on the type of reservoir.

```
if(v.gt.0) then
```

Note: For a discovered field primary development is started but constrained based on technology penetration and the maximum fraction of reservoirs allowed to be developed. The code accounts for Federal Lands Technology Penetration Curve. 'Dtcpenyx' is the effective development penetration rate.

```
dtcpenyx =  dtcpen(y,x)*(1-frac_fed(ii)) +
(fedpend(y,x)+dtcpen(y,x))*frac_fed(ii)

vscl=amin1(vscl,(dtcpenyx-kdox(wi,v)))

vscl=amin1(vscl,(kdomx(v)-kdo(v)))
vscl=amax1(vscl,0.0)
yfld=vscl*xfld
kdo(v)=kdo(v)+vscl
kdox(wi,v)=kdox(wi,v)+vscl

elseif(u.gt.0) then
```

Note: For secondary development on a reservoir the fraction of wells by secondary development option is obtained. Development is constrained based on technology penetration and the fraction of development already decided on. The code accounts for Federal Lands Technology Penetration Curve. 'Dtcpenyx' is the effective development penetration rate.

```
call udopk(1,udox(d,wi,u),udoxb)
```

```
        dtcpenyx =  dtcpen(y,x)*(1-frac_fed(u)) +
        (fedpend(y,x)+dtcpen(y,x))*frac_fed(u)

        vscl=amin1(vscl,(dtcpenyx-udoxb(b)))
        vscl=amin1(vscl,(1.0-udo(b,d,u)))
        vscl=amax1(vscl,0.0)
        yfld=vscl*xfld
```

Note: The code loops through the secondary development options and assigns the development of current option to all stages. The development arrays (udo,udox) are set up so that (udo(b)) equals the amount of development in the current option through all remaining options.

```
        b1=b
1020    vscla=vscl

        dtcpenyx =  dtcpen(y,x)*(1-frac_fed(u)) +
        (fedpend(y,x)+dtcpen(y,x))*frac_fed(u)


        vscl=amin1(vscl,(dtcpenyx-udoxb(b1)))
        vscl=amin1(vscl,(1.0-udo(b1,d,u)))
        vscl=amax1(vscl,0.0)

        vscla=vscla-vscl
        if(vscla.gt.0.0) then
        do 1025 a2=1,a1
        a3=bs(a2)
        if(bpnt(a3).eq.(mxnefl+u)) then
        bd3=btec(a)/(mxndty+1)
        w3=btec(a)-bd3*(mxndty+1)
        x3=xdeqx(w3,rsty)
        d3=(bd3-1)/3
        b3=bd-d3*3
        if((b3.eq.b1).and.(d3.eq.d).and.(x3.eq.x)) then
        xdrlmn=xdrlmn+bdrl(a3)*vscla
        go to 1026
        endif
        endif
1025    continue
1026c   continue
        endif
        udoxb(b1)=udoxb(b1)+vscl
        udo(b1,d,u)=udo(b1,d,u)+vscl
        b1=b1-1
        if(b1.gt.0) go to 1020
        if((udoxb(2).lt.udoxb(3)).or.(udoxb(1).lt.udoxb(2))) then
        vxx2=amax1(0.0,udoxb(3)-udoxb(2))
        udoxb(2)=amax1(udoxb(2),udoxb(3))
        vxx1=amax1(0.0,udoxb(2)-udoxb(1))
        udoxb(1)=amax1(udoxb(1),udoxb(2))
        if(wi.eq.1) then
        iwi=2
        else
        iwi=1
        endif
```

```
call udopk(1,udox(d,iwi,u),udoxc(1))
udoxc(1)=udoxc(1)-vxx1
udoxc(2)=udoxc(2)-vxx2
vxx2=amax1(0.0,udoxc(3)-udoxc(2))
udoxc(2)=amax1(udoxc(2),udoxc(3))
vxx1=amax1(0.0,udoxc(2)-udoxc(1))
udoxc(1)=amax1(udoxc(1),udoxc(2))
if((vxx1.gt.0.001).or.(vxx2.gt.0.001)) call errmsg(3,996)

call udopk(2,udox(d,iwi,u),udoxc)
endif

call udopk(2,udox(d,wi,u),udoxb)

elseif((g.gt.0).and.(iexpf.eq.0)) then
```

Note: For the initial development of discovered/undeveloped reservoir the breakeven point on rig capacity available is located. (should no longer be used)

```
dtcpenyx =  dtcpen(y,x)*(1-frac_fed(g)) +
(fedpend(y,x)+dtcpen(y,x))*frac_fed(g)

vscl=amin1(vscl,(dtcpenyx-gdox(wi,g)/xfld))
vscl=amin1(vscl,(gdomx(g)-gdo(g)))
vscl=amax1(vscl,0.0)
yfld=xfld*vscl
yfld=amin1(yfld,(plybfl(f,p)-gdo(g)))
gdo(g)=gdo(g)+yfld
gdox(wi,g)=gdox(wi,g)+yfld

elseif((f.gt.0).and.(p.gt.0).and.(iexpf.eq.1)) then
```

Note: Exploration is constrained based on technology penetration.

```
etcpenyz =    etcpen(y,z)*(1-frac_fed(g)) +
(fedpene(y,z)+etcpen(y,z))*frac_fed(g)

vscl=amin1(vscl,(etcpenyz*availpen(y,xa,s)
-fpdoz(z,g)/xfld))
vscl=amax1(vscl,0.0)
yfld=xfld*vscl
if(etcflf(f,f,z).le.0) call errmsg(3,997)
```

Note: The results of exploration in terms of number of reservoirs found per successful well by size class is determined.

```
vlflft=0.0
vlfo=0.0
do 1033 f1=1,nfsz(p)
vlfn=amax1(0.0,plynfl(f1,p)-vlfo)
vlfo=vlfn*2.0
```

```
        if(f1.ge.f) then
        if(f1.eq.f) vlfn=amax1(vlfn,0.001)
        vlflf(f1)=etcflf(f1,f,z)*vlfn
        if(etcflfp(f1,f1,z,p).gt.0.0)
        vlflf(f1)=etcflfp(f1,f,z,p)*vlfn
        else
        vlflf(f1)=0.0
        endif
1033    continue
        do 1032 f1=nfsz(p),f,-1
        if(f1.gt.f) then
        vlflf(f1)=vlflf(f1)+2.0*vlflf(f1-1)
        endif
        vlflft=vlflft+vlflf(f1)
1032    continue
        vlflft=(1.0+2.0)/vlflft
        do 1034 f1=f,nfsz(p)
        vlflf(f1)=vlflf(f1)*vlflft
1034    continue
```

Note: The maximum is reset based on the number of reservoirs remaining in largest size class. The number of fields found by exploration is saved. 'Fpdo' is set to the number of reservoirs found by size class and 'fpdoz' is set to the number of exploration steps implemented by technology.

```
        yfld=amin1(yfld,(plynfl(f,p)-fpdo(g))/vlflf(f))
        do 1035 f1=f,nfsz(p)
        g1=nnflp(f1,p)
        if((g1.gt.0).and.(g1.le.nnfl)) then
        fpdo(g1)=amin1((fpdo(g1)+yfld*vlflf(f1)),
        plynfl(f,p))
        endif
1035    continue
        fpdoz(z,g)=fpdoz(z,g)+yfld
        else
        call errmsg(4,604)
        endif

        endif
```

Note: The breakeven drilling cost is reset.

```
        bsrfl(a)=bsrfl(a)*(drlfcy/drlfcs)
```

Note: The drilling cost variables are updated to incorporate the reservoir decision.

```
        if((yfld.gt.0.0).and.(iexpf.eq.0)) then
```

Note: For the development drilling reservoir, 'bdrl(a)' is the drilling per well, 'yfld' is the number of wells, and 'xdrlmn' is the credit received for secondary drilling options that override previous secondary drilling selections.

'valn' is set to drilling after update
'valo' is set to drilling before update
'valx' is set to drilling capacity after maximum capacity additions

```
valn=drldrg(s,y)+bdrl(a)*yfld-xdrlmn
valo=drldrg(s,y)
valx=drtdrg(s,y)*xchx
```

Note: The amount of spare capacity from other regions used is updated and the total drilling in region and total drilling nationally is reset.

```
drunt=drunt+(amax1(valn,valx)-amax1(valo,valx))
drunt=amax1(drunt,drxnt)
drldrg(s,y)=valn
drldnt(y)=drldnt(y)+bdrl(a)*yfld-xdrlmn
```

Note: The current marginal drilling cost is re-estimated.

```
valv=drtdrg(s,y)*xchv
valf=drtdrg(s,y)*xchf
valt=drtdrg(s,y)*xchx
if(valn.le.valv) then
drcdrg(y,s)=drlvcx
elseif(valn.le.valf) then
drcdrg(y,s)=(valn-valv)/(valf-valv)*(drlfcx-drlvcx)+drlvcx
elseif(valn.le.valt) then
drcdrg(y,s)=drlfcx
else
if((drxnt.gt.0.0).and.(jfl.eq.0)) then
drcdrg(y,s)=drlfcx+drunt/drxnt*drltcs
else
drcdrg(y,s)=drlfcx+drltcs
if(drcdrg(y,s)/drlfcs.gt.2.0)drcdrg(y,s)=drlfcx
endif
endif
if (jfl.eq.0) vsvd(s)=drcdrg(y,s)
elseif((yfld.gt.0.0).and.(iexpf.ne.0)) then
```

Note: For the exploration drilling reservoir, 'bdrl(a)' is the drilling per well, 'yfld' is the number of wells, and 'xdrlmn' is the credit received for secondary drilling options that override previous secondary drilling selections.

'valn' is set to drilling after update
'valo' is set to drilling before update
'valx' is set to drilling capacity after maximum capacity additions

```
        valn=drlerg(s,y)+bdrl(a)*yfld
        valo=drlerg(s,y)
        valx=drterg(s,y)*xchx
```

Note: The amount of spare capacity from other regions used is updated and the total drilling in region and total drilling nationally is reset.

```
        drunt=drunt+(amax1(valn,valx)-amax1(valo,valx))
        drunt=amax1(drunt,drxnt)
        drlerg(s,y)=valn
        drlent(y)=drlent(y)+bdrl(a)*yfld
```

Note: The current marginal drilling cost is re-estimated.

```
        valv=drterg(s,y)*xchv
        valf=drterg(s,y)*xchf
        valt=drterg(s,y)*xchx
        if(valn.le.valv) then
        drcerg(y,s)=drlvcx
        elseif(valn.le.valf) then
        drcerg(y,s)=(valn-valv)/(valf-valv)*(drlfcx-drlvcx)+drlvcx
        elseif(valn.le.valt) then
        drcerg(y,s)=drlfcx
        else
        if((drxnt.gt.0.0).and.(jfl.eq.0)) then
        drcerg(y,s)=drlfcx+drunt/drxnt*drltcs
        else
        drcerg(y,s)=drlfcx+drltcs
        if(drcerg(y,s)/drlfcs.gt.2.0)drcerg(y,s)=drlfcx
        endif
        endif
        if (jfl.eq.0) vsve(s)=drcdrg(y,s)
        endif
        endif
1090    continue
        call errmsg(1,927)
```

Step 14: **At this point the reservoirs have been evaluated. If this was the first pass through the reservoirs (jfl=0) then the amount of drilling capacity that can be transported to other regions is estimated. The amount of transported capacity used, the factors used to calculate capacity at different price levels and the marginal cost of drilling to full plus transported costs are reset, i.e., in this step variables are rest based on actual drilling occurred.**

```
        if(jfl.ne.0) go to 1200
        jfl=1
        drxnt=0.0
        do 1092 s=1,nsrg
        vx1=amax1(drldrg(s,y),drtdrg(s,y)*drlchf/100.0)
        vx2=amax1(drlerg(s,y),drterg(s,y)*drlchf/100.0)
        drxnt=drxnt+
        amax1(0.0,(( (drtdrg(s,y)+drterg(s,y))*
        xchx-(vx1+vx2) )*dr_cap(s)) )
1092    continue
        drunt=0.0
        xchv=0.0
        xchf=0.0
        xchx=0.0
        xrrt=0.0
        do 1095 s=1,nsrg
        drltcs = dr_rig(s)
        vsvd(s)=drcdrg(y,s)
        vsve(s)=drcerg(y,s)
        drcdrg(y,s)=drlfcx+drltcs
        drcerg(y,s)=drlfcx+drltcs
1095    continue

        go to 1010
```

Step 15: **The marginal drilling costs before allowing transport of capacity are saved.**

```
1200    do 1210 s=1,nsrg
        drcdrg(y,s)=vsvd(s)
        drcerg(y,s)=vsve(s)
1210    continue
        endif
        call errmsg(1,928)
        call gett(tmes(9),tmea(9),1)
```

Step16: **The code goes through list of discovered reservoirs and resets the development program specifications to include decisions already made. This step updates variables based on what exactly occurred in the year so that next year calculations can be done.**

```
        call gett(tmes(10),tmea(10),0)
        u=upfrst
1500    do 1590 v=1,nefl
```

Note: The code screens to see if there is any activity on the reservoir.

```
        if(kdo(v).gt.0.0) then
```

Note: The reservoir type and the total number of wells available for development by pay grade (tnwl(d)) are obtained. The variable val is used to force complete development if only a fraction of the wells are left to develop.

```
        rsty=kdrsty(v)
        val=0.0
        do 1510 d=1,mxngr
        tnwl(d)=kdnwla(d,v)
        val=val+tnwl(d)
1510    continue
        if(val.le.0.1) then
        val=1.0/kdo(v)
        else
        val=1.0
        endif
```

Note: The code loops through the technology options (current, advanced), to get the technology type, and determines if development has been decided with this technology (kdox(wi,v)>0).

```
        do 1520 wi=1,mxndtx
        x=xdeqx(wi,rsty)
        kdox(wi,v)=kdox(wi,v)*val
        if(kdox(wi,v).gt.0.0) then
```

Note: The development program is added to the list (the list is packed if necessary).

```
        if(nuds.ge.mxnuds) then
        call pknuds(1,u)
        endif

        nuds=nuds+1
```

Note: The screening flags are initialized and the pointer is set to the discovered reservoir.

```
        do 1512 wj=1,mxndtx
        uflg(wj,nuds)=0
1512    continue
        upcde(nuds)=v
```

Note: Wells are assigned by pay grade and development arrays are set so that the model does not immediately assign secondary development to the program.

```
        do 1515 d=1,mxngr
        upnwl(d,nuds)=tnwl(d)*kdox(wi,v)
        kdnwla(d,v)=kdnwla(d,v)-upnwl(d,nuds)
        if(kdnwla(d,v).le.0.001) kdnwla(d,v)=0.0
        udo(1,d,nuds)=0.0
        udo(2,d,nuds)=0.0
        udo(3,d,nuds)=0.0
        do 1514 wj=1,mxndtx
        udox(d,wj,nuds)=0
1514    continue
1515    continue
```

Note: The start year and technology is saved and it is specified that no secondary development is assigned yet.

```
        upyr(nuds)=tmex+y-1
        upsyr(nuds)=0
        uptchp(nuds)=x
        uptchs(nuds)=0
        upsect(nuds)=0
```

Note: Pointers are updated to include development programs.

```
        call insu(nuds,u,upcde,upf,upb,upfrst,uplast)
        endif
1520    continue
        endif
1590    continue
        call errmsg(1,929)
        call gett(tmes(10),tmea(10),1)
```

Step 17: **The code goes through the list of development programs and updates the status of secondary development based on actual development decisions conducted.**

Note: At first the development decision is obtained from the list.

```
        if(nuds.gt.0) then
        u=upfrst
```

Note: It is verified that for a specific selected development program some action is being taken with a pay grade. Scaling factors (vscld) are set up to force complete action if only a small fraction of wells is left to develop.

```
1700    val=0.0
        do 1710 d=1,mxngr
        val=val+udo(1,d,u)
        tnwl(d)=upnwl(d,u)
        do 1705 b=1,mxdevt
        vala=(1.0-udo(1,d,u))*tnwl(d)
        if((vala.le.0.01).and.(udo(1,d,u).gt.0.0)) then
        vscld(b)=1.0/udo(1,d,u)
        else
        vscld(b)=1.0
        endif
1705    continue
1710    continue
```

Note: This piece of code tests to see if some action is being taken (val >0) and if so pointers to development field (g or v) and reservoir type are obtained.

```
        if(val.gt.0.0) then
        if(upcde(u).le.nefl) then
        v=upcde(u)
        rsty=kdrsty(v)
        else
        g=upcde(u)-nefl
        rsty=ndrsty(g)
        endif
```

Note: The code loops through technology options (current, advanced) and for secondary development options with the technology to generate the development programs for each technology and secondary development option.

```
nudsv = nuds
do 1780 wi=1,mxndtx
x=xdeqx(wi,rsty)
do 1770 b=1,mxdevt
```

Note: The code loops through all of the pay grades to see if the development reservoir applies to all of the wells covered in the original development program (jfl is set to 1 if not) and to estimate the total number of wells (val) that the development reservoir applies to.

```
        jfl=0
        val=0.0
```

```
      do 1715 d=1,mxngr
      call udopk(1,udox(d,wi,u),udoxb)
      if(b.le.2) then
      udoxb(b)=udoxb(b)-udoxb(b+1)
      endif
      udoxb(b)=udoxb(b)*vscld(b)
      val=val+udoxb(b)
      vala=tnwl(d)-udoxb(b)*upnwl(d,u)
      if(vala.gt.0.0) jfl=1
1715  continue
```

Note: The processing continues if the number of wells that the development reservoir applies to under the specified technology is greater than zero.

```
      if(val.gt.0.0) then
```

Note: If the development reservoir does not apply to all of the wells in the current development program then the current development program is split with one program including all the wells that the development reservoir applies to and the other including the remainder of the wells. The new development program gets all of the wells that have the secondary development applied to them.

```
      if(jfl.eq.1) then
```

Note: The development programs are packed if the decision number is higher than the maximum allowed.

```
      if(nuds.ge.mxnuds) then
      call pknuds(1,u)
      endif
```

Note: Here, new programs are created, the screening flags are initialized, and the pointers are saved to the discovered/undeveloped reservoir.

```
      nuds=nuds+1
      do 1716 wj=1,mxndtx
      uflg(wj,nuds)=0
1716  continue
      upcde(nuds)=upcde(u)
```

Note: The code loops through pay grades and determines the number of wells that the development reservoir applies to for the specified technology and secondary development option.

```
        do 1720 d=1,mxngr
        call udopk(1,udox(d,wi,u),udoxb)
        if ((udoxb(b).lt.0.0).or.(udoxb(b).gt.1)) then
        write(*,*)'nuds,udoxb(1),(2),(3),b='
        write(*,*)nuds,udoxb(1),udoxb(2),udoxb(3),b
        end if
        if(b.le.2) then
        udoxb(b)=udoxb(b)-udoxb(b+1)
        endif
        udoxb(b)=udoxb(b)*vscld(b)

        upnwl(d,nuds)=udoxb(b)*upnwl(d,u)
```

Note: The number of wells remaining to be assigned is updated and the development reservoir specifications for the new (split) development program are initialized.

```
        if (upnwl(d,nuds).gt.tnwl(d)) upnwl(d,nuds) = tnwl(d)

        tnwl(d)=tnwl(d)-upnwl(d,nuds)

        if (tnwl(d).lt.0.0) then

        write(*,*)'tnwl(d),upnwl(d,u),udoxb(1),
        udoxb(2),udoxb(3),b'
        write(*,*) 'vscld(1),vscld(2),vscld(3)'

        write(*,*)tnwl(d),upnwl(d,u),udoxb(1),
        udoxb(2),udoxb(3),b
        write(*,*) vscld(1),vscld(2),vscld(3)

        end if

        udo(1,d,nuds)=0.0
        udo(2,d,nuds)=0.0
        udo(3,d,nuds)=0.0
        do 1718 wj=1,mxndtx
        udox(d,wj,nuds)=0
1718    continue
1720    continue
```

Note: The starting year for primary and secondary development, the technologies used, and the secondary development options selected are saved.

```
        upyr(nuds)=upyr(u)
        upsyr(nuds)=tmex+y-1
        uptchp(nuds)=uptchp(u)
        uptchs(nuds)=x
        upsect(nuds)=b
        ui=u
```

Note: The new (split) development program is inserted into the sorted list.

```
        call insu(nuds,ui,upcde,upf,upb,upfrst,uplast)
        else
```

Note: The development reservoir is applied to all wells. The starting year, technology, and secondary development option are saved.

```
        upsyr(u)=tmex+y-1
        uptchs(u)=x
        upsect(u)=b

        endif
        endif
1770    continue
1780    continue
```

Note: The number of wells on the original development program are reset to those left after all of the program splits.

```
        do 1785 d=1,mxngr
        upnwl(d,u)=tnwl(d)

1785    continue
        endif
```

Note: The next development program is obtained.

```
        u=upf(u)
        if(u.ne.0) go to 1700
1790    continue
        endif
        call errmsg(1,931)
```

Step 18: **The code loops through all exploration prospects, summarizes the exploration activities for the year and updates the number of undiscovered and undeveloped prospects.**

Note: The code loops through the number of plays and field size classes in each play.

```
        do 1890 p=1,nply
        s=plyrga(p)
        if(nfsz(p).gt.0) then
```

```
      do 1880 f=1,nfsz(p)
```

Note: The pointer to field description is obtained and it is verified if any activity for the exploration step has been selected.

```
      g=nnflp(f,p)
      if(fpdo(g).gt.0.0) then
```

Note: If there is some activity the code loops through the exploration technologies to see if there is any activity for an exploration step with a corresponding selected technology.

```
      do 1870 z=1,netc
      if(fpdoz(z,g).gt.0.0) then
```

Note: The level of activity is obtained and the mix of reservoirs found with the exploration step is determined.

```
      valmx=fpdoz(z,g)
      yfld=0.0
      vlflft=0.0
      vlfo=0.0
      do 1814 f1=1,nfsz(p)
      vlfn=amax1(0.0,plynfl(f1,p)-vlfo)
      vlfo=vlfn*2.0
      if(f1.ge.f) then
      if(f1.eq.f) vlfn=amax1(vlfn,0.001)
      vlflf(f1)=etcflf(f1,f,z)*vlfn
      if(etcflfp(f1,f1,z,p).gt.0.0)
      vlflf(f1)=etcflfp(f1,f,z,p)*vlfn
      else
      vlflf(f1)=0.0
      endif
1814  continue
      do 1817 f1=nfsz(p),f,-1
      if(f1.gt.f) then
      vlflf(f1)=vlflf(f1)+2.0*vlflf(f1-1)
      endif
      vlflft=vlflft+vlflf(f1)
1817  continue
      vlflft=(1.0+2.0)/vlflft
      do 1816 f1=f,nfsz(p)
      vlflf(f1)=vlflf(f1)*vlflft
1816  continue
```

Note: The code loops through reservoir sizes and assigns the reservoirs that have been found.

```
        do 1820 f1=f,nfsz(p)
        xfld=amin1(plynfl(f1,p),valmx*vlflf(f1))
```

Note: Here the last segment of the reservoir found is obtained.

```
        if(f1.eq.f)then
        if(plynfl(f1,p).le.0.25.and.xfld.gt.0.0)xfld=plynfl(f1,p)
        endif

        yfld=yfld+xfld
        if(xfld.gt.0.0) then
        plybfl(f1,p)=plybfl(f1,p)+xfld
        plynfl(f1,p)=plynfl(f1,p)-xfld
        endif
1820    continue
```

Note: The summary arrays of exploration drilling (spledr), exploration non-drilling costs (sploec), exploration drilling costs (spledc), and the number of exploration wells (splenw) are updated.

```
        yfld=yfld/mxngr
        spledr(y,p)=spledr(y,p)+yfld/etcsrt(f,z,p)*
        (etcdrl(z,p)/drleff)/1000.0
        sploec(y,p)=sploec(y,p)+
        yfld/etcsrt(f,z,p)*etccst(z,p)/1000.0
        call drlcst(2,etcdrl(z,p),drlfcy,s)
        spledc(y,p)=spledr(y,p)*drcerg(y,s)*(drlfcy/drlfcs)
        splenw(y,p)=splenw(y,p)+yfld/etcsrt(f,z,p)
        endif
1870    continue
        endif
1880    continue
        endif
1890    continue
        call errmsg(1,932)
        call gett(tmes(10),tmea(10),1)
```

Step 19: **Here the year loop ends and the development programs arrays are packed.**

```
        u=1
        if(nuds.gt.0) call pknuds(1,u)

5000    CONTINUE    !END THE LARGE YEAR LOOP
        y=nyr+1
```

Step 20: **The routine ends.**

```
        return
        end
```

SUBROUTINE PKNUDS

CALLED BY:	EXDVST (Decides which exploration and development options will be selected each year.) EXDVSO (The reservoir shut-in decisions are made and the output variables are calculated.)
CALLS:	ERRMSG (Prints out errors and warnings) INSU (Updates the pointers for the development programs.)
MAIN THEME:	This routine packs the array specifications of the development programs and saves the data no longer required in the exploration and production module to a temporary binary file.

Step 1: **If ‘ic’ equals zero then the temporary file ‘upsave.tmp’ is opened and the flags are initialized.**

```
      if(ic.eq.0) then
      call errmsg(1,971)
990   format(1x,'pknuds -ic:',i4)
```

Note: The size of records in temporary file is defined.

```
      nn=2*6+4*mxngr
      open(46,file='upsave.tmp',form='BINARY',recl=nn)
      upflag=0
```

Step 2: **If ‘ic’ equals one then the data is no longer required for processing.**

Note: The data is saved in the temporary files and the output file is packed.

```
      elseif(ic.eq.1) then
      call errmsg(1,972)
```

Step 3: **The screening is initially set only to allow development programs with all their secondary development complete to be stored in a temporary file.**

Note: The minimum number of wells for the next pass is initialized.

```
      jfl=0
100   valmn=999999999.9
```

Step 4: **The code loops through the development programs beginning with the first development program (uj) in the list.**

Note: Packing is not allowed if there is only one development program is in the list.

```
110   uj=upfrst
      if(nuds.le.1) go to 190
```

Step 5: **The pointers for the previous development program in the list (ub) and the next development program in the list (uf) are obtained.**

Note: The current pointer (uk) is saved. The screening flag is tested for the type of decision needed for removing the development program from the list of selected reservoirs.

```
120   ub=upb(uj)
      uf=upf(uj)
      uk=uj
```

Note: The following 'if' statement indicates that if there are no wells to be assigned secondary development then the following steps are carried over.

```
      if(jfl.eq.0) then
```

Step 6: **If there is secondary development of a reservoir (such as re-fraced, infill, denoted by the 'upsyr' variable) then the removal flag (kfl) is set to a non zero number (i.e., one) indicating that secondary development is already done and hence no further secondary development will be done.**

```
      if(upsyr(uj).ne.0) then
```

Step 7: **If there is no secondary development then the 'kflg' flag is set to one indicating that secondary development could be done in the future if the economics of the reservoir improve.**

```
kfl=1
else
```

Step 8: **Here checks are performed and pointers and resource types are obtained.**

Note: For example, 'g1' is an undiscovered pointer and 'v1' is a discovered pointer. 'rsty' is the resource type for the reservoir.

```
kfl=0
if(upcde(uj).gt.nefl) then
```

Step 9: **Here pointers for undiscovered reservoirs are obtained.**

```
g1=upcde(uj)-nefl
v1=0
rsty=ndrsty(g1)
else
```

Step 10: **Here pointers for discovered reservoirs are obtained.**

```
v1=upcde(uj)
g1=0
rsty=kdrsty(v1)
endif
```

Step 11: **The technology used for primary development (uptchp(uj)), 'wj' (the index for the technology resource combination type) and the index of the year that primary development started (y1)are obtained.**

```
x1=uptchp(uj)
wj=xdeqv(x1,rsty)+1
y1=upyr(uj)-tmex+1
```

Step 12: **The code goes through the technology resource combinations and checks to see if the reservoir is undiscovered or discovered.**

Note: 'y3' which is the variable which indicates the year when secondary development would be done is assigned accordingly. Whenever the secondary development year is greater than (and equal to) the current year (y) the control loops out of the '122 do' loop.

```
      do 122 wj=1,mxndty
      if(xdeqy(1,wj,rsty).eq.x1) then
      if(upcde(uj).gt.nefl) then
      y3=y1+ndwin(wj,g1)
      else
      y3=y1+kdwin(wj,v1)
      endif
      if(y3.ge.y) go to 123
      endif
122   continue
```

Step 13: **If all secondary development decisions are made then the development program is marked for removal indicating no future secondary development option is available.**

```
      kfl=1
 123  continue
```

Note: This 'endif' indicates the end of the 'if' statement from Step 4a.

```
      endif
```

Step 14: **Initially 'val' is set to zero indicating that no wells have undergone secondary development and hence no wells are removed from the list of prospective options.**

Note: 'val' is the number of wells in the reservoir that are assigned for secondary development.

```
      val=0.0
```

Note: This 'else' indicates the default option for the 'if' statement in Step 3c.

```
      else
```

Step 15: **From the 2nd pass onwards (i.e., when 'ifl' is non-zero), the total number of wells remaining for development is calculated and is used to determine if these wells can be removed from the list of prospective options available in the year.**

```
      val=0.0
      do 125 dj=1,mxngr
      val=val+upnwl(dj,uj)
125   continue
```

Step 16: **If the calculated number of wells is less than the screening flag (Ifl i.e., the actual number of wells that can be used for secondary development) the decision is marked for removal from the prospective options and is included for activity in the current year.**

Note: i.e., if 'val' is less than or equal to 'ifl' then no secondary development would be done in the current year and drilling decisions have to wait for future years to be checked.

```
      if(val.le.jfl) then
      kfl=1
      else
      kfl=0
      endif
      endif
```

Step 17: **If 'kfl' is equal to zero, i.e., secondary development is conducted then the minimum number of wells assigned for secondary development is used.**

```
      if(kfl.eq.0) then
      valmn=amin1(valmn,val)
      endif
```

Step 18: **If the development program is marked for removal (i.e., 'kfl' = 1) it is removed.**

```
      if((kfl.eq.1).and.(uj.ne.ux)) then
```

Step 19: **The flag is set to indicate that the data is output and the data is stored in the temporary file (upsave.tmp).**

```
upflag=1
write(46) upcde(uj),(upnwl(dj,uj),dj=1,mxngr),upyr(uj),
upsyr(uj),uptchp(uj),uptchs(uj),upsect(uj)
```

Step 20: **The pointers are reset so that the previous and next development decisions point to each other and the index of the first and last development decisions are correct.**

Note: The pointer to the current program (uk) is set to the previous development program so that the current loop will continue with the next development program.

```
if(ub.gt.0) then
upf(ub)=uf
uk=ub
else
upfrst=uf
uk=0
endif
if(uf.gt.0) then
upb(uf)=ub
else
uplast=ub
endif
```

Step 21: **If the removed development decision is not the last in the array then the last development decision in the array is moved to the emptied spot.**

```
if(uj.ne.nuds) then
```

Step 22: **If necessary the pointer is reset to the current development program.**

```
if(ux.eq.nuds) ux=uj
```

Step 23: **The specifications are moved so that they are not available from next year onwards for the reservoir being processed.**

```
      upcde(uj)=upcde(nuds)
      do 140 wi=1,mxndtx
      uflg(wi,uj)=uflg(wi,nuds)
140   continue
      do 150 dj=1,mxngr
      upnwl(dj,uj)=upnwl(dj,nuds)
      udo(1,dj,uj)=udo(1,dj,nuds)
      udo(2,dj,uj)=udo(2,dj,nuds)
      udo(3,dj,uj)=udo(3,dj,nuds)
      do 145 wi=1,mxndtx
      udox(dj,wi,uj)=udox(dj,wi,nuds)
145   continue
150   continue
      upyr(uj)=upyr(nuds)
      upsyr(uj)=upsyr(nuds)
      uptchp(uj)=uptchp(nuds)
      uptchs(uj)=uptchs(nuds)
      upsect(uj)=upsect(nuds)
```

Step 24: The pointers are reset to the previous and next development programs.

```
      uf=upf(nuds)
      ub=upb(nuds)
      upb(uj)=ub
      if(ub.gt.0) then
      upf(ub)=uj
      else
      upfrst=uj
      endif
      upf(uj)=uf
      if(uf.gt.0) then
      upb(uf)=uj
      else
      uplast=uj
      endif
      endif
```

Step 25: The number of development programs in the arrays is reduced since one is assigned in earlier steps.

```
      nuds=nuds-1
      endif
```

Step 26: The pointer to the next development program is obtained and the removal process is continued if appropriate.

```
      if(uk.gt.0) then
      uj=upf(uk)
      else
      uj=upfrst
      endif
      if((uj.ne.0).and.(nuds.gt.1)) go to 120
```

Step 27: **The code checks to see if enough space is made available and if not the screening flag (jfl) is reset to a value greater than the minimum number of wells not yet processed.**

Note: The code then loops back to control statement ‘100’.

```
190   if(nuds.le.mxnuds*0.75) go to 200
      jfl=max1(float(jfl+1),valmn*1.1)
      go to 100
```

Step 28: **A check is performed on the maximum number of decisions that can be allowed in the exploration and production run.**

```
200   if(nuds.ge.mxnuds) then
      call errmsg(4,605)
      endif
```

Note: If ‘ic’ equals two then the data is retrieved from the temporary file (upsave.tmp).

```
      elseif(ic.eq.2) then
      call errmsg(1,973)
```

Step 29: **The current pointer and number of development programs is initialized.**

```
      if(upflag.ne.1) then
      call errmsg(1,974)
      nuds=0
      endif
      uj=nuds
```

Step 30: **The status flag (upflag) is tested and if the data has already been written then the file is closed and the flag is set to two indicating that the data can be read if needed.**

```
      if(upflag.eq.1) then
      call errmsg(1,975)
      endfile 46
      rewind 46
      upflag=2
      endif
```

Step 31: If there is data in the file and the file is ready to be read then the records are read from the temporary file until the end of file is reached or the arrays are full.

```
      if(upflag.eq.2) then
      if(nuds.lt.mxnuds) then
      call errmsg(1,976)
250   uj=uj+1
      read(46,end=300) upcde(uj),(upnwl(dj,uj),dj=1,mxngr),upyr(uj),
      upsyr(uj),uptchp(uj),uptchs(uj),upsect(uj)
      do 260 wi=1,mxndtx
      uflg(wi,uj)=0
260   continue
      nuds=uj
      if (uj.ge.32760) then
      endif
      if(nuds.lt.mxnuds) go to 250
      endif
```

Step 32: If the number of records is less than the maximum then all input from the temporary file is complete and the status of the flag is reset to indicate that no data is available.

```
300   if(nuds.lt.mxnuds) upflag=0
```

Step 33: The first record is initially set up as the first in the list.

```
      upfrst=1
      uplast=1
      upf(1)=0
      upb(1)=0
      uj=1
```

Step 34: The pointers for all the remaining records are set.

```
      if(nuds.gt.1) then
      call errmsg(1,977)
      do 350 ui=2,nuds
      if (ui.ge.32760) then
      endif
      call insu(ui,uj,upcde,upf,upb,upfrst,uplast)
```

```
350   continue
      endif
      endif
      endif
```

Step 35: The subroutine ends.

```
      return
      end
```

SUBROUTINE PRJSRT

CALLED BY: EXDVST (Decides which exploration and development options will be selected each year.)
PRJPAK (Packs the project arrays.)

CALLS: GETT (Estimates the time required for each phase of the subroutine.)

MAIN THEME: This routine sorts the development projects in descending order of economical drilling cost.

Step 1: The number of records (n) and the initial comparison increment (m) is assigned.

```
        n=nprj
        m=n
```

Step 2: The comparison increment is reduced by 2 and the code checks to see if sorting needs to be done.

```
104     m=m/2
        if(m.le.0) go to 200
```

Step 3: The number of comparison groups is obtained and the loop begins.

```
        l=n-m
        do 109 j=1,l
        i=j+m
        106  i=i-m
        if(i.le.0) go to 109
        i1=bs(i)
        i2=bs(i+m)
```

Step 4: Sorting is performed based on the indices obtained.

```
        if(bsrfl(i1).gt.bsrfl(i2)) go to 109
        if((bsrfl(i1).eq.bsrfl(i2)).and.(i1.lt.i2)) go to 109

        bs(i)=i2
        bs(i+m)=i1
        go to 106
```

```
109     continue
        go to 104

200     continue
```

Step 5: The subroutine ends.

```
        call gett(tmes(15),tmea(15),1)
        return
        end
```

SUBROUTINE PRJPAK

CALLED BY: EXDVST (Decides which exploration and development options will be selected each year.)

CALLS: ERRMSG (Prints out errors and warnings)
PRJSRT (Sorts the development projects.)
UDOPK (Packs/unpacks specification of secondary development options.)

MAIN THEME: This routine packs the project arrays by making decisions on secondary development options that do not require any drilling used only for recomplete case.

Step 1: **Here the projects are sorted in descending order of breakeven drilling cost.**

```
call prjsrt
```

Step 2: **The number of items processed and that can be deleted is initialized. The code then loops through all of the projects in order.**

```
nn=0
do 200 i=1,nprj
j=bs(i)
```

Step 3: **It is ensured that the project is secondary development and that the drilling required is 0 (this is for refracs only).**

```
if((bpnt(j).gt.mxnefl).and.(bpnt(j).le.(mxnefl+mxnuds))) then
if(bdrl(j).le.0.0) then
```

Step 4: **The specifications of secondary option, technology, and pay grade are obtained. The pointer is set to the development program.**

```
bdz=btec(j)/(mxndtc+1)
wz=btec(j)-bdz*(mxndtc+1)
dz=(bdz-1)/3
bz=bdz-dz*3
```

```
        uz=bpnt(j)-mxnefl
```

Step 5: **The current specification of selected development is obtained for the development program and preparations are made to update the utilization.**

```
        vscl=1.0
        call udopk(1,udox(dz,wz,uz),udoxb(1))
        b1=bz
```

Step 6: **The utilization is updated so that the report utilization for an option is included in the utilization arrays for that option and all preceding options.**

Note: The use of the option by technology penetration is constrained.

```
100     vscl=amin1(vscl,(dtcpen(y,wz)*availpen(y,wz,s)-udoxb(bz)))
        vscl=amin1(vscl,(1.0-udo(bz,dz,uz))) !check for not higher than 100%
        vscl=amax1(vscl,0.0)                !check for net less than 0%
```

Note: The utilization is updated.

```
        udoxb(b1)=udoxb(b1)+vscl
        udo(b1,dz,uz)=udo(b1,dz,uz)+vscl
```

Note: Here the code proceeds to the next option.

```
        b1=b1-1
        if(b1.gt.0) go to 100
```

Note: The utilization array is repacked and the project is marked as deleted.

```
        call udopk(2,udox(dz,wz,uz),udoxb(1))
        bpnt(j)=0
        nn=nn+1
        endif
        endif
200     continue
```

Step 7: **The sorted order is updated and the deleted projects are moved to the end of the project list.**

Note: The number of projects remaining and the order of going through the list of projects is initialized.

```
        i2=nprj
        i=0
```

Note: The code loops through all the remaining development projects.

```
205     i=i+1
210     if(i.ge.i2) go to 300
        j=bs(i)
```

Note: The code skips over projects that have not been deleted.

```
        if(bpnt(j).ne.0) go to 205
```

Note: The project is deleted and the sort order of remaining projects is updated. The deleted project is put at the end of the list and the number of projects remaining is revised.

```
        do 220 i3=i,i2-1
        bs(i3)=bs(i3+1)
        220  continue
        bs(i2)=j
        i2=i2-1
        go to 210
300     continue
```

Note: It is ensured that the proper number was deleted from the sort order.

```
        if(i2.ne.(nprj-nn)) call errmsg(3,503)
```

Step 8: **The arrays are packed.**

Note: The number of records after packing (ix), the sort index of the first project (i1) available to be moved up physically in the list and the sort index of the first project to be deleted from the list (i2) are set up.

```
        ix=i2
        i1=1
        i2=nprj
```

Note: If the sort indices for the next available project to move up and the next project to delete are the same then the packing process is terminated. The code jumps out of loop.

```
310     if(i1.ge.i2) go to 500
```

Note: The index of next project for deletion is obtained. If it is not a deleted project then the loop terminates.

```
        j2=bs(i2)
        if(bpnt(j2).ne.0) go to 500
```

Note: Here the code checks to see if the project to be deleted is physically located before the spot where the last non-deleted project will be and if it is then the code continues, otherwise it skip over project to the next one on the list and tries again.

```
        if(j2.le.ix) go to 350
        i2=i2-1
        go to 310
```

Note: If there is a deleted project in the arrays where we want to pack to the code searches for a non-deleted project in the latter part of the arrays that can be moved up. The loop terminates if we get to a deleted project.

```
350     j1=bs(i1)
        if(bpnt(j1).eq.0) go to 500
        if(j1.gt.ix) go to 400
        i1=i1+1
        go to 310
```

Note: Here there is a match and the non-delete project is moved up in the arrays while the sort indices are reset.

```
400     bpnt(j2)=bpnt(j1)
        breg(j2)=breg(j1)
        btec(j2)=btec(j1)
        bnfl(j2)=bnfl(j1)
        bsrfl(j2)=bsrfl(j1)
        bdelt(j2)=bdelt(j1)
        bdrl(j2)=bdrl(j1)
        bpnt(j1)=0
        bs(i1)=j2
        bs(i2)=j1
```

Note: The code skips to the next deleted project.

```
        i2=i2-1
        go to 310
```

Note: Consistency checks are performed.

```
500     if(ix.ne.i2) then
        write(*,501) i1,i2,bpnt(bs(i1)),bpnt(bs(i2)),ix,nn,nprj
501     format(' ',7i5)
        call errmsg(4,502)
        endif
```

Note: The number of projects is updated.

```
        nprj=ix
```

Step 9: **The subroutine ends.**

```
        return
        end
```

SUBROUTINE INSU

CALLED BY: EXDVST (Decides which exploration and development options will be selected each year.)
PKNUDS (Packs the arrays containing the specifications of the development programs and saves the data no longer required to temporary files.)

MAIN THEME: This routine updates the pointers for the development decisions so that they are always in order. It also inserts a new development decisions into the appropriate place in the list if needed.

Step 1: **Here the reservoir pointer for the inserted development decision is compared to that of the last one on the list.**

Note: If the reservoir pointer is less than that of the last one on the list then the code loops backward through the list until the one that has a reservoir pointer less than or equal to the one being inserted is found. If the code gets to the beginning of the list then it stops and sets the inserted one as the first.

```
      100 if(upcde(ui).ge.upcde(u)) go to 200

      if(u.ne.upfrst) go to 150
      upfrst=ui
      upf(ui)=u
      upb(ui)=0
      upb(u)=ui
      go to 400

150   u=upb(u)
      go to 100
```

Step 2: Here the code checks to see if it is at the end of the list and if so the new development decision is inserted at the end of the list.

```
200   if(u.ne.uplast) go to 250
      uplast=ui
      upf(u)=ui
      upf(ui)=0
      upb(ui)=u
      go to 400
```

Step 3: **If the code is not at the end of the list it loops through the list until the end is found or it reaches a development decision with a pointer that is greater than the one being inserted.**

```
250     un=upf(u)
        if(upcde(ui).lt.upcde(un)) go to 300
        u=un
        go to 200
```

Step 4: **The development decision is inserted at this point on the list. The forward and backward pointers are set to reflect the insertion.**

```
300     upf(u)=ui
        upf(ui)=un
        upb(ui)=u
        upb(un)=ui
```

Step 5: **The subroutine ends.**

```
400     continue
        if (ui.gt.32760) then
        endif
        return
        end
```

Table of Contents

SUBROUTINE DVLMSP

CALLED BY: EXDVST (This routine decides which exploration and development options will be selected each year.)

CALLS: GETT (Estimates the time required for each phase of the subroutine.)
ERRMSG (Prints out errors and warnings)
DRLCST (Estimates drilling cost for the play based on the drilling depth.)

MAIN THEME: This routine estimates the MASP and MASP parameters for a reservoir at contemporary conditions (i.e., at current gas price, technology penetration and market conditions).

Step 1: **Calls the 'GETT' subroutine to estimate time elapsed in the 'DVLMSP' routine and converts the discount rate into a fraction.**

```
Call gett(tmes(12),tmea(12),0)
disc=disrte/100.
```

Step 2: **This code decodes input corresponding to a secondary development on a development program to the basic reservoir pointers.**

Note: The appropriate pointers to the reservoirs are obtained and saved into 'j' along with the resource type 'icx'.

```
if((ic.eq.1).or.(ic.eq.3)) then  !ic = 3 und., ic = 1 disc.
j=i
icx=ic
elseif(ic.eq.2) then
if(upcde(i).le.nefl) then
icx=1
j=upcde(i)
else
icx=3
j=upcde(i)-nefl
endif
else
call errmsg(4,501)
endif
```

Step 3: **The number of development options (1 - for primary and 3 for secondary) are initialized.**

Note: The specific data items required to estimate MASP from the discovered or new/banked reservoir arrays is obtained and the number of development options is set.

```
if(jc.eq.0) then
b1=1
else
b1=mxdevt
endif

IF (icx.eq.1) THEN
```

Step 4: **The following section of code assigns parameters for discovered reservoirs.**

Note: Here, depth (dpth), price (prc), resource type (rsty), supply region (sx), play counter (px) are assigned.

```
dpth=kddpth(j)
px=kdflpp(j)
rsty=kdrsty(j)
sx=plyrga(px)
prc=supprc(yi,sx)
```

Note: The following 'do loop' (loop 50) assigns entries from the reservoir data-bank file to useable variables.
Values for the drilling slope (xpvds), non drilling slope (xpvns), non drilling cost (for environmental regulations,(xpvndb)), compliance cost due to drilling activity (for environmental regulation, (xpvdcb)) and infill drilling costs for additional wells (if 'b2' equals three) are calculated.
'xpvndb' is the total expenses at the current market price, 'xpvdcb' is total investment at the current market price, and 'xpvtxb' is total taxes at the current market price.

```
do 50 b2=1,b1
xpvp(b2)=kdpvp(b2,di,wi,j)
frac_pp(b2)=frac_p(b2,di,wi,j)
xpvndb(b2)=kdpvnd(b2,di,wi,j)+(prc-2.00)* kdpcnd(b2,di,wi,j)
xpvdcb(b2)=kdpvdc(b2,di,wi,j)+(prc-2.00)* kdpcdc(b2,di,wi,j)
xpvtxb(b2)=kdpvtx(b2,di,wi,j)+(prc-2.00)* kdpctx(b2,di,wi,j)
xpvds(b2)=kdpvds(b2,di,wi,j)
xpvns(b2)=kdpvns(b2,di,wi,j)
if(ienv.ge.1)then
xpvndb(b2)=xpvndb(b2)+envndnc(ienv,j)* kdnwl(1,di,j)
xpvdcb(b2)=xpvdcb(b2)+envdcnc(ienv,j)* kdnwl(1,di,j)
```

```
        +envdcf(ienv,j)*dpth*kdnwl(1,di,j)
        do iyre=1,kdpryr(b2,di,wi,j)
        xpvndb(b2)=xpvndb(b2)+
        kdnwl(1,di,j)*envexnc(ienv,j)/(1+disc)**(iyre-1)
        enddo
        if(b2.eq.3)then
        iwin=kdwin(di,j)
        xpvndb(b2)=xpvndb(b2)+
        kdnwl(1,di,j)*envndnc(ienv,j)/(1+disc)**iwin
        xpvdcb(b2)=xpvdcb(b2)+
        kdnwl(1,di,j)*(envdcnc(ienv,j)+envdcf(ienv,j)*dpth)
        /(1+disc)**iwin
        do iyre=iwin,kdpryr(b2,di,wi,j)
        xpvndb(b2)=xpvndb(b2)+envexnc(ienv,j)*kdnwl(1,di,j)
        /(1+disc)**(iyre-1)
        enddo
        endif
        xpvndb(b2)=xpvndb(b2)+xpvp(b2)*envgc(ienv,j)
        +(envwc(ienv,j)*watyld(j)*xpvp(b2))
        endif

50      continue

        xnw=kdnwl(4,di,j)

        ELSE
```

Step 5: **The following section of code assigns parameters for undiscovered reservoirs.**

Note: Here, depth (dpth), price (prc), resource type (rsty), supply region (sx), play counter (px) are assigned.

```
        dpth=kddpth(j)
        px=kdflpp(j)
        rsty=kdrsty(j)
        sx=plyrga(px)
        prc=supprc(yi,sx)
```

Note: The following ‘do loop’ (loop 60) assigns entries from the reservoir data-bank file to useable variables.
Values for the drilling slope (xpvds), non drilling slope (xpvns), non drilling cost (for environmental regulations,(xpvndb)), compliance cost due to drilling activity (for environmental regulation, (xpvdcb)) and infill drilling costs for additional wells (if ‘b2’ equals three) are calculated.
‘xpvndb’ is the total expenses at the current market price, ‘xpvdcb’ is total investment at the current market price, and ‘xpvtxb’ is total taxes at the current market price.

```
        dpth=nddpth(j)
```

```
px=nnflb(j)/mxnfsz
rsty=ndrsty(j)
sx=plyrga(px)
prc=supprc(yi,sx)
do 60 b2=1,b1
xpvp(b2)=ndpvp(b2,di,wi,j)
xpvndb(b2)=ndpvnd(b2,di,wi,j)+(prc-2.00)*ndpcnd(b2,di,wi,j)
xpvdcb(b2)=ndpvdc(b2,di,wi,j)+(prc-2.00)*ndpcdc(b2,di,wi,j)
xpvtxb(b2)=ndpvtx(b2,di,wi,j)+(prc-2.00)*ndpctx(b2,di,wi,j)
xpvds(b2)=ndpvds(b2,di,wi,j)
xpvns(b2)=ndpvns(b2,di,wi,j)
if(ienv.ge.1)then
xpvndb(b2)=xpvndb(b2)+envndnc(ienv,j)*ndnwl(di,j)
xpvdcb(b2)=xpvdcb(b2)+envdcnc(ienv,j)*ndnwl(di,j)

do iyre=1,ndpryr(b2,di,wi,j)
xpvndb(b2)=xpvndb(b2)+
ndnwl(di,j)*envexnc(ienv,j)/(1+disc)**(iyre-1)
enddo
if(b2.eq.3)then
iwin=ndwin(di,j)
xpvndb(b2)=xpvndb(b2)+
ndnwl(di,j)*envndnc(ienv,j)/(1+disc)**iwin
xpvdcb(b2)=xpvdcb(b2)+
ndnwl(di,j)*envdcnc(ienv,j)/(1+disc)**iwin
do iyre=iwin,ndpryr(b2,di,wi,j)
xpvndb(b2)=xpvndb(b2)+
ndnwl(di,j)*envexnc(ienv,j)/(1+disc)**(iyre-1)

enddo
endif
xpvndb(b2)=xpvndb(b2)+xpvp(b2)*envgc(ienv,j)
+envwc(ienv,j)*watyld(j)*xpvp(b2)
endif
60  continue
xnw=ndnwl(di,j)

ENDIF
```

Step 6: **The technologies used for primary development (x1) and secondary development (x2) are stored from reservoir databank file entries (xdeqy).**

Note: It is assumed that primary and secondary development types would use the same technology type.

```
x1=xdeqy(1,wi,rsty)
x2=xdeqy(2,wi,rsty)

do 70 b2=1,b1

xpvtxb(b2)=txfndr(hi)*xpvndb(b2)+txfdrl(hi)*xpvdcb(b2)+
txfmrg(hi)*xpvp(b2)*prc+xpvtxb(b2)

xpvds(b2)=xpvds(b2)-txfdrl(hi)
xpvns(b2)=xpvns(b2)-txfndr(hi)
```

Step 7: **The variable non-drilling cost (xpvndv) gets calculated based on non-drilling cost decline factor.**

Note: The variable drilling cost (xpvdcv) is calculated based on drilling cost decline factor (tchd) up to that year, and the variable to full drilling cost ratio (i.e., drlvcs/drlfcs which is generally 81% - These values are specified in the file 'drl_cap.spc.'

```
        xpvndv(b2)=xpvndb(b2)*dtccsf(yi,x1)
        if(drlfcs.eq.0.0) call errmsg(3,813)

        xpfdc2(b2)=xpvdcb(b2)
        xpvdcv(b2)=xpvdcb(b2)*tchd*drlvcs/drlfcs
```

Step 8: **Variable tax calculations are performed.**

Note: These variable entries (drilling costs, non-drilling costs and taxes) are the minimum values for the corresponding entries.

```
        xpvtxv(b2)=xpvtxb(b2)+(xpvndb(b2)-xpvndv(b2))*(1-xpvns(b2))+
        (xpvdcb(b2)-xpvdcv(b2))*(1-xpvds(b2))
```

Note: If the reservoir is on federal lands the taxes ('xpvtxv' variable which includes both royalty and taxes) are adjusted for royalty relief.

```
        if (hi.ge.1) then
        xpvtxv(b2) = xpvtxv(b2) - xpvp(b2)*(royrate(j)-roy_incentive)*
        frac_fed(j)* frac_p(b2,di,wi,j)*prc
        endif

70      continue
```

Step 9: **If the variable drilling cost factor 'drlvcs' or 'tchd' (actual drilling cost decline factor) are zero, the program prints a fatal error message and stops.**

```
        if(jc.eq.0) then
        prda=xpvp(1)
        if(prda.gt.0.0) then

        if(drlvcs.eq.0.0) call errmsg(3,871)
        if(tchd.eq.0.0) call errmsg(3,872)
```

Note: If the development well success rate (plydsc) is zero (i.e., the ‘ply_dfn.spc’ file is zero) then the rate is assigned to be 80%.

```
if(plydsc(wi,px).eq.0.0) then
write(*,9931) px,plynme(px),ic,j,nnflb(j),
plydsc(wi,px)
9931    format(' px: ',i3,a20,3i10/10(1x,f9.3))
call errmsg(3,873)
plydsc(wi,px)=0.8
endif
```

Note: For primary wells this code computes the MASP and the variable drilling cost. The MASP component also includes the gas processing cost in $/Mcf.

```
mspo(1,1)=(xpvndv(1)+xpvdcv(1)+xpvtxv(1))/prda
+proc_cst(1,j)
```

Note: Here the ‘drlcst’ routine is called to obtain the $/ft cost for the reservoir which is then multiplied by depth (dpth) and the number of wells (xnw) to get well cost. The variable gets divided by the success rate to get the total drilling cost including unsuccessful wells.

```
call drlcst(1,dpth,drlfcy,sx)
if(xnw.gt.0.0) then
vddd=drlfcy*dpth/plydsc(wi,px)/drlfcs*xnw
else
vddd=.0001
endif
```

Step 10: The change in the total cost (per MCF for primary development) for each dollar change in drilling costs is calculated below.

```
mspo(2,1)=((xpfdc2(1)-xpvdcv(1))* xpvds(1))/((drlfcs-drlvcs*tchd)
*vddd)/prda
```

Note: ‘mspo(3,1)’ is the total footage needed to be drilled taking into account unsuccessful development drilling activities.

```
mspo(3,1)=dpth/plydsc(wi,px)
if(xnw.gt.0.0) then
```

Note: ‘msp4(1)’ is the (primary well) production per well in BCF/year/well.

```
mspo4(1)=prda/xnw
else
mspo4(1)=prda
endif
else
```

Step 11: If production is zero from the reservoir data bank file then various variables are assigned specific values.

Note: A very high value (9999.9) is assigned to the MASP, a minimal value (0.1) is assigned to the change in total cost per dollar change in drilling cost, and a negligible value (0.001) is assigned to the production per well.

```
mspo(1,1)=9999.9
mspo(2,1)=0.1
if(plydsc(wi,px).eq.0.0)then
call errmsg(3,802)
plydsc(wi,px)=0.8
endif
mspo(3,1)=dpth/plydsc(wi,px)
mspo4(1)=0.001
endif

endif
```

Step 12: Calculations for secondary development such as re-frac, infill cases are performed.

```
if(jc.ne.0) then

x1=xdeqy(1,wi,rsty)
x2=xdeqy(2,wi,rsty)
```

Step 13: The incremental NPV production (prda) is computed for refrac case and further calculations are performed when refrac case gives a higher production than the primary case.

```
prda=xpvp(2)-xpvp(1)
if(prda.gt.0.0) then
```

Note: If the non-drilling cost decline factor (value specified in the 'dtec_pen.spc' file) is zero an error message is printed on the screen.

```
if(dtccsf(yi,x1).eq.0.0) call errmsg(3,803)
```

Note: 'Val' is the difference between refrac and primary case non-drilling cost adjusted by cost decline factors.

```
val=(xpvndv(2)-xpvndv(1))*(dtccsf(yi,x2)/ dtccsf(yi,x1)-1.0)*
xpvns(2)
```

Note: 'mspo(1,1)' is the increment from the primary case to the refrac case.

```
mspo(1,1)=((xpvndv(2)+xpvdcv(2)+xpvtxv(2))-
(xpvndv(1)+xpvdcv(1)+xpvtxv(1))+val)/prda
+ proc_cst(1,j)
```

Note: 'mspo(2,1)' is the change in the total cost (per MCF for primary development for each dollar change in drilling costs.

```
mspo(2,1)=0.00000000001
```

Note: 'mspo(3,1)' is the total footage needed to be drilled taking into account unsuccessful development drilling activities.

```
mspo(3,1)=0.0
if(xnw.gt.0.0) then
```

Note: 'msp4(1)' is the (primary well) production per well in BCF/year/well.

```
mspo4(1)=prda/xnw
else
mspo4(1)=prda
endif
else
```

Step 14: **If production is zero from the reservoir data bank file then various variables are assigned specific values.**

Note: A very high value (9999.9) is assigned to the MASP, a minimal value (0.1) is assigned to the change in total cost per dollar change in drilling cost, a value of zero is assigned to the total footage needed to be drilled taking into account unsuccessful development drilling activities and a negligible value (0.001) is assigned to the production per well.

```
mspo(1,1)=9999.9
mspo(2,1)=0.1
mspo(3,1)=0.0
mspo4(1)=0.001
endif
```

Step 15: The incremental NPV production (prda) is computed for infill case and further calculation are performed when infill case gives a higher production than the primary case.

```
prda=xpvp(3)-xpvp(1)
if(prda.gt.0.0) then
```

Note: If the non-drilling cost decline factor is zero an error message is printed on the screen.

```
if(dtccsf(yi,x1).eq.0.0) call errmsg(3,804)
```

Note: 'Val' is the difference between refrac and primary case non-drilling cost adjusted by cost decline factors.

```
val=(xpvndv(3)-xpvndv(1))*(dtccsf(yi,x2)/dtccsf(yi,x1)-1.0)*
xpvns(3) !change in d.c. as change in tech
```

Note: 'mspo(1,1)' is the increment from the primary case to the refrac case.

```
mspo(1,2)=((xpvndv(3)+xpvdcv(3)+xpvtxv(3))-
(xpvndv(1)+xpvdcv(1)+xpvtxv(1))+val)/prda
+ proc_cst(1,j)
if(drlvcs.eq.0.0) call errmsg(3,881)
if(tchd.eq.0.0) call errmsg(3,882)
if(plydsc(wi,px).eq.0.0)then
call errmsg(3,805)
plydsc(wi,px)=0.80
endif
```

Note: Here the drilling cost routine (drlcst) is called to get the infill well cost

```
call drlcst(1,dpth,drlfcy,sx)
if(xnw.gt.0.0) then
vddd=drlfcy*dpth/plydsc(wi,px)/drlfcs*xnw
else
vddd=.0001
endif
```

Note: 'mspo(2,2)' is the change in the total cost (per MCF for primary development for each dollar change in drilling costs.

```
mspo(2,2)=((xpfdc2(3)-xpfdc2(1))-(xpvdcv(3)-xpvdcv(1)))
*xpvds(3)/((drlfcs-drlvcs*tchd)
*vddd)/prda
```

Note: 'mspo(3,2)' is the total footage needed to be drilled taking into account unsuccessful development drilling activities.

```
mspo(3,2)=dpth/plydsc(wi,px)
if(xnw.gt.0.0) then
```

Note: 'msp4(2)' is the (primary well) production per well in BCF/year/well.

```
mspo4(2)=prda/xnw
else
mspo4(2)=prda
endif

else
```

Step 16: **If production is zero from the reservoir data bank file then various variables are assigned specific values.**

Note: A very high value (9999.9) is assigned to the MASP, a minimal value (0.1) is assigned to the change in total cost per dollar change in drilling cost, and a negligible value (0.001) is assigned to the production per well.

```
mspo(1,2)=9999.9
mspo(2,2)=0.1
if(plydsc(wi,px).eq.0.0)then
call errmsg(3,806)
plydsc(wi,px)=0.8
endif
```

```
mspo(3,2)=dpth/plydsc(wi,px)
mspo4(2)=0.001
endif
```

Step 17: If infill and refrac do not exist, they have no impact.

```
mspo(1,3)=9999.9
mspo(2,3)=0.1
if(plydsc(wi,px).eq.0.0)then
call errmsg(3,807)
plydsc(wi,px)=0.8
endif
mspo(3,3)=dpth/plydsc(wi,px)
mspo4(3)=0.001

endif
```

Step 18: The routine ends.

```
call gett(tmes(12),tmea(12),1)

RETURN
END
```

SUBROUTINE GETT

CALLED BY: DVLMSP.FOR (Estimates the MASP parameters for a selected project.)
EXDVI4 (Reads in input production specifications.)
EXDVST.FOR (Decides which exploration and development decision will be selected each year.)
EXPLPROD.FOR (Controls the main flow through the expl./dvlp. model.)
PRJSRT.FOR (Sorts the development projects.)
EXDVI2.FOR (Reads data bank files and performs checks.)

CALLS: GETTIM (Microsoft Fortran Utility routine that computes time elapsed)

MAIN THEME: This routine estimates the amount of time required in each phase of the routine. It may be used for tuning the model and speeding it up.

Step 1: **The system time is obtained and converted to 100ths of a second.**

Note: 'ihr' is an integer, represents the hour, and can take on values from zero to 23 (gets the value from the system clock). 'imin' is an integer, represents the minutes, and can take on values from zero to 59 (gets the value from the system clock). 'isec' is an integer, represents the second, and can take on values from zero to 59 (gets the value from the system clock). 'i100th' is an integer, represents a hundredth of a second, and can take on values from zero to 99 (gets the value from the system clock). Actual arguments used in calling 'gettim' must be integers, array elements or structure elements. The 'ttmp' variable provides the value in integers.

```
call gettim(ihr,imin,isec,i100th)
ttmp=(((float(ihr)*60.0)+float(imin))*60.0+float(isec))*100.0+
float(i100th)
```

Step 2: **Based on the control code, either the current time is saved or the net time since last time is calculated and saved.**

```
if(icd.eq.0) then
tme=ttmp
```

```
else
tacc=tacc+(ttmp-tme)
endif
```

Step 3: The subroutine ends.

```
return
end
```

Table of Contents

SUBROUTINE EXDVSO

CALLED BY: EXPLPROD (Controls the main flow through the expl./dvlp. model.)

CALLS: PKNUDS (Packs the arrays containing the specifications of the development programs and saves the data no longer required to temporary files.)
EXDVI3 (Reads in input production specifications from binary files.)
EXDVI4 (Reads in input production specifications from binary files.)
DRLCST (Estimates drilling cost for the play based on the drilling depth.)
GETT (Estimates the time required for each phase of the subroutine.)
ERRMSG (Prints out errors and warnings)

CREATES: 'price.out' (Contains gas prices, drilling cost factors, number of 'expl. wells' and average depth of exploration wells drilled.)
'decision.out' (Contains a summary of investment decisions.)
'prodsumm.out' (Contains supply production summary report.)
'resvsumm.out' (Contains reserve summary report.)
'suppsumm.out' (Contains supply summary by region and resource type.)
'wellsumm.out' (Contains exploration wells, development wells and operating wells.)
'explwls.out' (Contains number of exploration wells drilled by play & year.)
'supply.ext' (Contains gas prices and supply in BCF.)

MAIN THEME: In this routine the reservoir shut-in decisions are made and the output variables are calculated. These are printed in appropriate output files.

Step1: **The price.out' file is written.**

Note: The 'Price.out' file is opened.

```
open(40,file='price.out')
```

Note: The discount rate 'disrte' is changed into a fraction.

```
        disc=disrte/100.
```

Note: The process of writing the price.out file is initiated. At first the name of the supply region (srgnme(s)) is written.

```
        do 2400 s=1,nsrg
        write(40,2301) srgnme(s)
2301    format(a20)
```

Note: Various local variables are initialized.

```
        Do y = 1, nyr
        spleny(y)=0.0
        totdep(y)=0.0
        Enddo
```

Note: The value of the number of exploratory wells drilled 'spleny' and total footage 'totdep' is assigned. This is done in the play loop 2310. For the supply region counter ‘s’, the number of exploratory wells for all plays are added together. In addition to the total footage drilled (totdep) is computed.

```
        do 2310 p=1,nply
        if(plyrga(p).eq.s) then
        do 2305 y=1,nyr
        spleny(y)=spleny(y)+splenw(y,p)
        totdep(y)=totdep(y)+splenw(y,p)*etcdep(p)
2305    continue
        endif
2310    continue
```

Note: Here the cost factors of development drilling (drcdrg) and exploratory drilling (drcerg) are assigned. The calendar year, ‘iy’ (1997, 1998, etc.), gas price at which the exploration and production module is run ‘supprc(y,s)’, the development (valxx) and exploration (valxy) drilling cost factors, number of exploration wells (spleny(y)), and average depth of exploration wells (totdep(y)/spleny(y)) are written into the price.out file.

```
        do 2350 y=1,nyr
        valxx=drcdrg(y,s)
        valxy=drcerg(y,s)
        if(valxy.lt.0.5)valxy=0.50
        if(valxy.gt.1.5)valxy=1.500
        iy=y+tmex-1
        if (spleny(y).le.0.0) then
        write(40,2302) iy,supprc(y,s),valxx,
        valxy,spleny(y),0.0
        else
        write(40,2302) iy,supprc(y,s),valxx,
        valxy,spleny(y),totdep(y)/spleny(y)
        endif
2302    format(i4,1x,f6.2,2(1x,f7.4),1x,f8.1,1x,f8.1)
2350    continue
2400    continue
```

Note: Finally, for all the resource types (both current and advanced technologies), i.e., 'dtcnme(x)', the relative cost of technology 'dtccsf(y,x)' is printed. These entries are the same as specified in the last column of the input file dtec_pen.spc.

```
        do 2500 x=1,ndtc
        write(40,2301) dtcnme(x)
        do 2450 y=1,nyr
        iy=y+tmex-1
        write(40,2402) iy,dtccsf(y,x)
2402    format(i4,1x,f7.3)
2450    continue
2500    continue
```

Note: The 'Price.out' file is closed.

```
        close(40)

        call errmsg(1,981)
        call pknuds(2,u)
        call errmsg(1,982)
```

Step2: **The remaining output files ('decision.out', 'prodsumm.out', 'resvsumm.out', 'suppsumm.out', 'wellsumm.out') are written.**

Note: In addition, shut-in decisions are performed on every reservoir (GSAMID).

Note: The following output files are opened: 'decision.out', 'prodsumm.out', 'resvsumm.out', 'suppsumm.out' and 'wellsumm.out'. In the suppsumm file, first the calendar year (e.g., 1997, 1998, etc.) is written. In addition if the ‘iocde’ variable is one or less, then the routine exdvi3 is called which creates the binary data bank. (This action is currently in-active). Since the files are already in binary format to start with. In addition the exdvi4 routine is called to get production and operation and maintenance cost response by year.

```
        open(40,file='decision.out')
        open(44,file='prodsumm.out')
        open(47,file='resvsumm.out')
        open(48,file='suppsumm.out')
        open(49,file='wellsumm.out')
        write(48,2331)(tme(t),t=1,ntme)
2331    format(25x,33(i5,5x))

        if(iocde.le.1) call exdvi3

        call exdvi4(0)
        call errmsg(1,983)
```

Note: Various local variables are initialized.

```
        do 3090 p=1,mxnplx
        do 3080 y=1,nyr
        spxprd(y,p)=0.0
        spxnpw(y,p)=0.0
3080    continue
3090    continue
```

Note: Here first local pointers are set. Then, the decision file (decision.out) is written.

```
4000    if(nuds.gt.0) then
        uf=upfrst

        do 4010 u=1,nuds
        if(u.ge.2)uf=upf(uf)

        i=upcde(uf)
        if(i.le.nefl) then
        p=kdflpp(i)
        f=0
        v=i
        else
        v=0
        p=nnflb(i-nefl)/ mxnfsz
        f=nnflb(i-nefl)-p*mxnfsz+1
        endif
        s=plyrga(p)
```

Note: The following variables are printed in the decision file:

column 1 = Decision # (uf)
column 2 = Reservoir Counter (upcde(uf))
column 3 = wells drilled in pay grade 1 (upnwl(d,uf))
column 4 = wells drilled in pay grade 2 (upnwl(d,uf))
column 5 = wells drilled in pay grade 3 (upnwl(d,uf))
column 6 = start year of primary development program (upyr(uf))
column 7 = year of secondary development program (0 means no sec. dev.) (upsyr(uf))
column 8 = index of technology for primary development (uptchp(uf))

- if index = 1 conventional gas reservoir (current)
- if index = 2 tight gas reservoir (current)
- if index = 3 radial flow gas reservoir, no fractures (current)
- if index = 4 linear flow gas reservoir, MHF (current)
- if index = 5 water drive reservoir (current)
- if index = 6 coal bed reservoir (current)
- if index = 7 offshore reservoir (current)
- if index = 8 conventional gas reservoir (advanced)
- if index = 9 tight gas reservoir (advanced)
- if index =10 radial flow gas reservoir, no fractures (advanced)
- if index =11 linear flow gas reservoir, MHF (advanced)
- if index =12 water drive reservoir (advanced)
- if index =13 coal bed reservoir (advanced)
- if index =14 offshore reservoir (advanced)

column 9 = index of technolgy for secondary development (uptchs(uf))
column 10 = secondary development type , 1:frac/refrac, 2:infill only, 3:infill with frac/refrac of current well (upsect(uf))

```
        write(40,4001) uf,upcde(uf),(upnwl(d,uf),d=1,mxngr),upyr(uf),
        upsyr(uf),uptchp(uf),uptchs(uf),upsect(uf)

4001    format(i6,1x,i6,1x,3f12.4,2(i5,1x),3(i2,1x))
4010    continue

        call errmsg(1,984)
        u=upfrst
        i=0
        p=0
```

Note: Here play variables dependent on year (in BCF) are converted to variables dependent on the year and the play (in TCF).

```
4050    if(i.gt.0) then
        do 4060 y=1,nyr
        splprd(y,p)=splprd(y,p)+fplprd(y)/1000.0
        spxprd(y,p)=spxprd(y,p)+fpxprd(y)/1000.0
        spldrl(y,p)=spldrl(y,p)+fpldrl(y)/1000.0
        spldcs(y,p)=spldcs(y,p)+fpldcs(y)/1000.0
        splndc(y,p)=splndc(y,p)+fplndc(y)/1000.0
        splwpr(y,p)=splwpr(y,p)+fplwpr(y)/1000.0
        splfom(y,p)=splfom(y,p)+fplfom(y)/1000.0
        splvom(y,p)=splvom(y,p)+fplvom(y)/1000.0
4060    continue
        endif
```

Note: 'fpl' variables are initialized.

```
        do 4080 y=1,nyr
        fplprd(y)=0.0
        fpxprd(y)=0.0
        fpldrl(y)=0.0
        fpldcs(y)=0.0
        fplndc(y)=0.0
        fplwpr(y)=0.0
        fplfom(y)=0.0
        fplvom(y)=0.0
4080    continue
```

Note: Various local variables are initialized.

```
        resvrp=0.0
        resvrs=0.0
        dvlnwp=0.0
        dvlnws=0.0
        dvlrfc=0.0
```

Note: The variables for depth (dpth) and resource type (rsty) are defined. In addition pointers for play (p) and field size class (f) get assigned.

```
        if(i.ne.upcde(u)) then
        i=upcde(u)
        call exdvi4(i)
        if(i.le.nefl) then
        p=kdflpp(i)
        f=0
        dpth=kddpth(i)
        rsty=kdrsty(i)
        prtyp(p)=rsty
        else
        p=nnflb(i-nefl)/ mxnfsz
        f=nnflb(i-nefl)-p*mxnfsz+1
        dpth=nddpth(i-nefl)
        rsty=ndrsty(i-nefl)
        call errmsg(4,714)
        endif
        s=plyrga(p)
        endif
```

Note: Here the primary year (y1) and the secondary year (y2) variables are stored.

```
        y1=upyr(u)-tmex+1
        if (y1.le.0) y1 = 1
        y2=upsyr(u)
        if(y2.ne.0) then
        jc=1
        y2=upsyr(u)-upyr(u)+1
        b=upsect(u)+1
        if(b.gt.3) call errmsg(4,715)
        else
        jc=0
        y2=mxnyr+1
        b=1
        endif
        x1=uptchp(u)
        x2=uptchs(u)
        if(jc.eq.0) x2 = x1
```

Note: Technology pointers are adjusted based on ‘x1’ and ‘x2’ as assigned earlier.

```
        do 4105 d=1,mxngr
        if(upnwl(d,u).gt.0.0) then
        wi=0
        ifomj(d)=0
        wp=0
        do 4100 wj=1,mxndty
        if((xdeqy(1,wj,rsty).eq.x1).and.(xdeqy(2,wj,rsty).eq.x2))
        wi=wj
        if((xdeqy(1,wj,rsty).eq.x1).and.(xdeqy(2,wj,rsty).eq.x1))
        wp=wj
4100    continue
        if((wi.eq.0).or.(wp.eq.0)) then
        write(*,4103) upcde(u),f,p,rsty,x1,x2,jc,y2
4103    format(' upcde,f,p,rsty,x1,x2,jc,y2: ',8i4)
        call errmsg(4,992)
        endif
4102    continue
        endif
4105    continue
```

Note: The environmental regulatory case is initialized and the environmental loop gets started for incorporating environmental costs in investment decisions.

```
ienv=0
do ipr=1,ienvcse
```

Note: The year for which regulations are specified is stored in the 'ienv' variable.

```
if(iyrenv(ipr).le.tme(ntme))then
if(y1+tmex-1.ge.iyrenv(ipr))ienv=ipr
endif
enddo

do 4200 y3=1,mxnyr-(y1+1)
y=y1+y3-1
h = taxcde(y)
y9=y+tmex-1
```

Note: The year is checked to see if the environmental regulations have changed. If regulations don't change in a year, last year's environmental compliance costs are used for the current year also.

```
if(y.gt.0) then
if(y9.ge.iyrenv(ienv+1).and.y9.lt.iyrenv(ienvcse+1))
ienv=ienv+1
if((y3.eq.1).and.(upyr(u).ge.tmex).and.(y.le.nyr)) then
do 4110 d=1,mxngr
if(upnwl(d,u).gt.0.0) then
infx=0.0
```

Note: 'fplndc' (the number of wells drilled in a reservoir in a paygrade stored for the current year), 'fpldrl' (the total footage drilled), 'fpldcs', 'dvlnwp' (the total number of wells drilled including dry development wells) are assigned values. If the development well success rate (plydsc) for the play is zero (as specified in the 'Ply_dfn.spc' file) it is assigned to 80%.

```
fplndc(y)=fplndc(y)+infx*upnwl(d,u)
call drlcst(1,dpth,drlfcy,s)
vddd=drlfcy/drlfcs
if(plydsc(wi,p).eq.0.0)then
call errmsg(3,802)
plydsc(wi,p)=0.8
endif
```

```
        fpldrl(y)=fpldrl(y)+dpth*upnwl(d,u)/
        plydsc(wi,p)
        fpldcs(y)=fpldrl(y)*drcdrg(y,s)*vddd
        if(plydsc(wi,p).eq.0.0)then
        call errmsg(3,802)
        plydsc(wi,p)=0.8
        endif
        dvlnwp=dvlnwp+upnwl(d,u)/plydsc(wi,p)
        endif
4110    continue
        endif
```

Note: In the exploration and production module the supply region is assigned based on the play. This region, play cross-walk is obtained from the ply_dfn.spc file.

```
        s=plyrga(p)
```

Note: Various local variables (‘fom’ : fixed operating and maintenance cost, ‘vom’ : variable operating and maintenance cost, and ‘envfom’ : environmental fixed operating and maintenance cost) are initialized.

```
        do 4120 d=1,mxngr
        fom=0.0
        vom=0.0
        envfom=0.0
```

Note: If the number of wells drilled is non zero (i.e., greater than 0.000001 in the code), then total production of the reservoir (prd), production of the primary case (prpd), and fixed operating and maintenance cost (foam) is calculated according to the number of wells drilled.

```
        if(upnwl(d,u).ge.0.000001) then
        prd=wprd(y3,b,d,wi)*upnwl(d,u)/kdnwl(4,d,i)
        prdp=wprd(y3,1,d,wp)*upnwl(d,u)/kdnwl(4,d,i)
        fom=(woam(y3,b,d,wi)*upnwl(d,u)/kdnwl(4,d,i)-vom*prd)/
        upnwl(d,u)
        fominit = fom*upnwl(d,u)
```

Note: The royalties and severance tax is adjusted for the gas price.

```
        adj_sev_roy = (2.0 - supprc(y,s))*prd*(royrate(i)+0.05)
        adj_oam = ( fominit-2.0*prd*(royrate(i) + 0.05) ) *
        0.2*(2.0-supprc(y,s))/2.
```

Note: The above factor (0.2*(2.0-supprc(y,s))/2.) is the operating and maintenance cost multiplier for gas price.

```
if (prd.le.0.0) then
adj_oam_mcf = 0.0
else
adj_oam_mcf = (adj_sev_roy+adj_oam)/prd
endif
fom = (fominit - adj_sev_roy - adj_oam)/upnwl(d,u)

vom=vom+proc_cst(1,i)
```

Note: If the total revenue (less variable operating and maintenance cost) is less than the operating and maintenance cost then the reservoir is not considered for investment purposes.

```
if (wprd(y3,b,d,wi)*(supprc(y,s)-vom+adj_oam_mcf).
lt.woam(y3,b,d,wi)) then
prd  = 0.0
prdp = 0.0
fom  = 0.0
go to 4219
end if
```

Note: This part of the code takes the environmental variables into consideration.

```
if(ienv.gt.0)then
iyre=iyrenv(ienv)-tmex+1
```

Note: Annual environmental cost is added in for new and existing wells.

```
if(iyre.le.y1)then
envfom = envexnc(ienv,i)*upnwl(d,u)
else
envfom = envexec(ienv,i)*upnwl(d,u)
endif

if(b.eq.3.and.y2.le.y3) then
```

Note: The environmental year is checked against the infill year, and if infill occurs extra compliance cost is added for the additional wells drilled. The environmental variable operating costs ('envgc' for gas, in $/MCF; ‘envwc’ for water, in $/bbl) are appropriately added to the original ‘vom’ to obtain the total ‘vom’ value.

```
if(iyre.le.y2+y1-1)then
envfom = envfom + envexnc(ienv,i)*upnwl(d,u)
else
envfom = envfom + envexec(ienv,i)*upnwl(d,u)
endif
endif
vom = vom+envgc(ienv,i)+envwc(ienv,i)* watyld(i)
```

Note: The cost is added for upgrading existing wells if the regulations for this year are new. The calculation is performed for the first year of production from the wells.

```
if(y.eq.iyre.and.y3.gt.1)then
prodleft=
-(envdcec(ienv,i)+envndec(ienv,i))
*upnwl(d,u)
```

Note: The calculation is performed if infill has occurred and is before current year.

```
if(b.eq.3.and.(y1+y2-1.lt.iyre))
prodleft=prodleft
-(envdcec(ienv,i)+envndec(ienv,i))
*upnwl(d,u)
```

Note: The cost is distributed to upgrade over the remaining productive life of the well. The extra royalty relief paid (royadd) is calculated in case of a royalty incentive run.

```
royadd = 0
do 1997 ilft = y3,mxnyr

if(prodleft.ge.0.0) goto 1998
if (taxcde(ilft+y1-1).ge.1) then
if (frac_fed(i).gt.0.0) then
if (wprd(ilft,b,d,wi)*1e03/kdnwl(4,d,i)/365.le.
rate_marg) then
royadd=(royrate(i)-roy_incentive)* wprd(ilft,b,d,wi)*
supprc(y,s)* frac_fed(i)
endif
endif
endif

if (wprd(ilft,b,d,wi)*(supprc(y,s)-vom+adj_oam_mcf)-
        woam(ilft,b,d,wi)+royadd.lt.0.0) goto 1997
        prodleft=prodleft+(wprd(ilft,b,d,wi)
        *upnwl(d,u)/ kdnwl(4,d,i)
        *(supprc(y,s)-vom+adj_oam_mcf)+royadd*upnwl(d,u)/ kdnwl(4,d,i)
        -envfom-(woam(ilft,b,d,wi)*upnwl(d,u)/ kdnwl(4,d,i)) )
        /(1+disc)**(ilft-y3)
1997    continue
1998    If(prodleft.lt.0.0) ifomj(d)=1
        endif
        endif

        foam=fom*upnwl(d,u)+vom*prd+envfom
```

Note: Total fixed operating and maintenance cost is adjusted for Canadian regions.

```
        if(s.eq.18.or.s.eq.19) foam=foam*exchrate
```

Note: This piece of code takes into account the shut in conditions.

```
        IF (h.ge.1) THEN
        if (frac_fed(i).gt.0.0.and.upnwl(d,u).gt.0.0) then
        if (prd*1e03/upnwl(d,u)/365.le.rate_marg) then
        foam = foam - (royrate(i)-roy_incentive)*prd*supprc(y,s)*
        frac_fed(i)
        endif
        endif
        ENDIF

        if ((prd*supprc(y,s)-foam).lt.0.0.or.ifomj(d).eq.1) then
        prd=0.0
        fom=0.0
        endif


        if((prdp*supprc(y,s)-foam).lt.0.0.or.ifomj(d).eq.1) then
        prdp=0.0
        endif

4219    vcs=fom*upnwl(d,u)+vom*prd
```

Note: Reserves in the primary case (resvrp) is calculated by summing production rates of the reservoir for all the years. Secondary reserves is calculated by summing the extra production obtained by infill and refrac cases for all the years. This section also calculates total wells drilled by year and play.

```
        resvrp=resvrp+prdp
        resvrs=resvrs+(prd-prdp)
        if(y.le.nyr) then
        if(prd.gt.0.0) then
        vscla=1.0
        if((b.eq.3).and.(y3.ge.y2)) then
        vscla=2.0
        endif
        splnpw(y,p)=splnpw(y,p)+upnwl(d,u)*vscla
        if(upyr(u).lt.tmex) then
        spxnpw(y,p)=spxnpw(y,p)+upnwl(d,u)
        fpxprd(y)=fpxprd(y)+prdp
        endif
        fplprd(y)=fplprd(y)+prd
        fplfom(y)=fplfom(y)+fom*upnwl(d,u)+envfom
        fplvom(y)=fplvom(y)+vom*prd+envgc(ienv,i)*prd+
        envwc(ienv,i)* watyld(i)*prd
        endif
        endif
        endif ! upnwl(d,u)>tolerance
4120    continue  ! pay grade
        iflag_p = iflag_p + 1
        upold = upcde(u)
        if((y3.eq.y2).and.(upsyr(u).ge.tmex).and.(y.le.nyr)) then
        do 4130 d=1,mxngr
        if(upnwl(d,u).gt.0.0) then
```

Note: Secondary investment costs are obtained.

```
        if(b.eq.2) then
        infx=0.0
        drl=0.0
        dvlrfc=dvlrfc+upnwl(d,u)
        else
        infx=0.0
        if(plydsc(wi,p).eq.0.0)then
        call errmsg(3,802)
        plydsc(wi,p)=0.8
        endif
        drl=dpth/plydsc(wi,p)
        dvlnws=dvlnws+upnwl(d,u)/plydsc(wi,p)
        endif
        fplndc(y)=fplndc(y)+infx*upnwl(d,u)
        fpldrl(y)=fpldrl(y)+drl*upnwl(d,u)
        call drlcst(1,dpth,drlfcy,s)
        vddd=drlfcy/drlfcs
        fpldcs(y)=fpldrl(y)* drcdrg(y,s)* vddd

        endif
4130    continue
        endif
        endif
4200    continue
```

Note: Detailed statistics are written as output. Reserve statistics are accumulated.

```
        y=upyr(u)-tmex+1
        if(y.lt.1) then
```

Note: The variables being assigned ('splirs': primary reserve addition array; 'spldwp' : total number of wells drilled) are being stored in array variables for the purpose of reporting.

```
      splirs(p)=splirs(p)+resvrp/1000.
      elseif(y.le.nyr) then
      splrsp(y,p)=splrsp(y,p)+resvrp/1000.
      spldwp(y,p)=spldwp(y,p)+dvlnwp
      endif
      y=upsyr(u)-tmex+1
      if(y.lt.1) then  !Commented to take into accounttmex
      splirs(p)=splirs(p)+resvrs/1000.
      elseif(y.le.nyr) then
      spldws(y,p)=spldws(y,p)+dvlnws
      splrss(y,p)=splrss(y,p)+resvrs/1000.
      splrfc(y,p)=splrfc(y,p)+dvlrfc
      endif
```

Note: The next project is obtained.

```
      u=upf(u)
```

Note: If activity does apply to same field then it is added into play statistics. The production, drilling etc. values by year (i.e., 'fplprd(y)', 'fpldrl(y)', etc.) are added for corresponding plays so that a play specific output could be created.

```
      if(u.gt.0) go to 4050
      if(i.gt.0) then
      do 4260 y=1,nyr
      splprd(y,p)=splprd(y,p)+fplprd(y)/1000.0
      spldrl(y,p)=spldrl(y,p)+fpldrl(y)/1000.0
      spldcs(y,p)=spldcs(y,p)+fpldcs(y)/1000.0
      splndc(y,p)=splndc(y,p)+fplndc(y)/1000.0
      splwpr(y,p)=splwpr(y,p)+fplwpr(y)/1000.0
4260  continue
      i=0
      endif
      endif
      if(nuds.le.0) goto 4300

      call pknuds(2,u)
      if(nuds.gt.0) go to 4000
```

Note: The supply region statistics and output play summaries are calculated. Various local variables are initialized. Variables ending in 'us' represent values for The United States, and values ending in 'cn' represent values for Canada.

```
        do t=1,ntme
        spsprdus(t)=0.0
        spsprdcn(t)=0.0

        spsenwus(t)=0.0
        spsdwpus(t)=0.0
        spsdwsus(t)=0.0
        spsnpwus(t)=0.0
        spsenwcn(t)=0.0
        spsdwpcn(t)=0.0
        spsdwscn(t)=0.0
        spsnpwcn(t)=0.0
        enddo
```

Note: The following variables are initialized:
Gas production (spsprd).
Exploratory wells drilled (spsenw).
Primary wells drilled (spsdwp).
Infill development wells drilled (spsdws).
Producing wells drilled (spsnpw)

Note: These variables are stored by time period (t) and supply region (s) specifications.

```
4300    do 5050 s=1,nsrg
        do 5040 t=1,ntme
        spsprd(t,s)=0.0

        spsenw(t,s)=0.0
        spsdwp(t,s)=0.0
        spsdws(t,s)=0.0
        spsnpw(t,s)=0.0
5040    continue
5050    continue
```

Note: The follow-up variables are stored by time period (t) and resource type (ir). The variables are: gas production (spsprdir), producing wells drilled (spsenwir), primary development wells drilled (spsdwpir), infill development wells drilled (spswsir) and producing wells drilled (spsnpwir).

```
        do 5070 ir=1,(2*mxrsty)
        do 5060 t=1,ntme
        spsprdir(t,ir)=0.0

        spsenwir(t,ir)=0.0
        spsdwpir(t,ir)=0.0
        spsdwsir(t,ir)=0.0
        spsnpwir(t,ir)=0.0
5060    continue
5070    continue
```

Note: The variable 'p' is assigned to the index of the supply region that the play is in (plyrga) and the variable 'spsprdy' (which is a variation of the variable 'spsprd') is set to zero.

```
        do p=1,mxnply
        do y=1,mxnyr
        s = plyrga(p)
        spsprdy(y,s) = 0.0
        enddo
        enddo
```

Note: For North Alaska total production (spsprdy) is hardwired to zero. The ‘spsprdy’ variable is the total production by year (y) and supply region (s).

```
        do p=1,mxnply
        do y=1,mxnyr
        s = plyrga(p)
        if (srgnme(s).eq.'North Alaska       ') spsprdy(y,s) = 0.0
        spsprdy(y,s)=spsprdy(y,s) + splprd(y,p)
        enddo
        enddo

        do 5150 p=1,nply
```

Note: Here the variables which are dependent on the time period and the supply region are assigned values based on the variables which are dependent on the year and play.

```
        s=plyrga(p)
        ir=rst(p)
        do 5130 t=1,ntme
        y=tme(t)-tmex+1
        spsprd(t,s)=spsprd(t,s)+splprd(y,p)
        spsenw(t,s)=spsenw(t,s)+splenw(y,p)
        spsdwp(t,s)=spsdwp(t,s)+spldwp(y,p)
        spsdws(t,s)=spsdws(t,s)+spldws(y,p)
        spsnpw(t,s)=spsnpw(t,s)+splnpw(y,p)
```

Note: The totals (by time period and resource type) are calculate in the following section for all the Canadian regions: Alberta, British Columbia, the MacKenzie Delta and Sable Island.

```
        if((srgnme(s).eq.'Alberta            ').or.
        (srgnme(s).eq.'British Columbia    ').or.
        (srgnme(s).eq.'MacKenzie Delta     ').or.
        (srgnme(s).eq.'Sable Island        '))then
        spsprdir(t,ir+mxrsty)=spsprdir(t,ir+mxrsty)+splprd(y,p)
        spsprdcn(t)=spsprdcn(t)+splprd(y,p)

        spsenwir(t,ir+mxrsty)=spsenwir(t,ir+mxrsty)+splenw(y,p)
        spsdwpir(t,ir+mxrsty)=spsdwpir(t,ir+mxrsty)+spldwp(y,p)
        spsdwsir(t,ir+mxrsty)=spsdwsir(t,ir+mxrsty)+spldws(y,p)
        spsnpwir(t,ir+mxrsty)=spsnpwir(t,ir+mxrsty)+splnpw(y,p)
        spsenwcn(t)=spsenwcn(t)+splenw(y,p)
        spsdwpcn(t)=spsdwpcn(t)+spldwp(y,p)
        spsdwscn(t)=spsdwscn(t)+spldws(y,p)
        spsnpwcn(t)=spsnpwcn(t)+splnpw(y,p)
        goto 5130
        end if
```

Note: If the supply region is in Mexico, the values are not added into the U.S. totals. U.S. totals are calculated by time period specifications and resource type.

```
        if (srgnme(s).eq.'Mexico-Supply       ') goto 5130
        spsprdir(t,ir)=spsprdir(t,ir)+splprd(y,p)
        spsprdus(t)=spsprdus(t)+splprd(y,p)

        spsenwir(t,ir)=spsenwir(t,ir)+splenw(y,p)
        spsdwpir(t,ir)=spsdwpir(t,ir)+spldwp(y,p)
        spsdwsir(t,ir)=spsdwsir(t,ir)+spldws(y,p)
        spsnpwir(t,ir)=spsnpwir(t,ir)+splnpw(y,p)
        spsenwus(t)=spsenwus(t)+splenw(y,p)
        spsdwpus(t)=spsdwpus(t)+spldwp(y,p)
        spsdwsus(t)=spsdwsus(t)+spldws(y,p)
        spsnpwus(t)=spsnpwus(t)+splnpw(y,p)

5130    continue
5150    continue
```

Note: The 'supply.ext' output file is opened and written. The header line (indicating years) are first written only for the first pass through the exploration and production module. The supply region name, the supply pass number, the well head price and total production by year and region is specified.

```
        open(43,file='supply.ext')
        if (iscase.eq.1) then
        write(43,5991) (y + tmex - 1,y+tmex-1,y=1,nyr)
5991    format(t26,33(i4,3x,i4,2x))
        endif

        do 5190 s=1,nsrg
        write(43,5171) srgnme(s),iscase,
        (supprc(y,s),spsprdy(y,s),y=1,nyr)
```

Note: The name of the supply region and the supplies by region and time period are written in the 'suppsumm.out' file.

```
        write(48,5172) srgnme(s),(spsprd(t,s),t=1,ntme)

5171    format(a20,i3,33(f6.2,f7.1))
5172    format(a20,33f10.1)
5173    format(/,'Total U.S.',10x,33f10.1,'//')
5190    continue

        write(48,5173)(spsprdus(t),t=1,ntme)
```

Note: The number of exploration wells drilled by play and by year are written in the 'explply.out' file

```
        open(444,file='explwls.out')
        write(444,4402) (y + tmex - 1,y=1,nyr)
        write(444,*) '            Wells Drilled ----> '
4402    format(' #  ', ' Play',40(4x,i4))
        do p=1,nply
        write(444,4401)p, plynme(p),(splenw(y,p),y=1,nyr)
        enddo
4401    format(i4,2x,a4,40f8.3)
        close(444)
```

Note: The supply summary by resource type and year is appended at the bottom of the 'suppsumm.out' file.

```
        do ir=1,(2*mxrsty)
        mir=ir
        if(mir.gt.mxrsty)mir=ir-mxrsty
        if(ir.eq.mxrsty+1) write(48,5174) (spsprdcn(t),t=1,ntme)
5174    format(/,'Total Canada',8x,11f10.1,'//')
        write(48,5172)rstynm(mir),(spsprdir(t,ir),t=1,ntme)
        enddo
```

Note: The 'wellsumm.out' file is created. This contains exploration wells, development wells and operating wells by time period (t) and by GSAM region. Also, the file contains exploration, development and operating wells by resource type and year.

```
        do 6001 fff=1,4
        if (fff.eq.1) write(49,6011)
        if (fff.eq.2) write(49,6012)
        if (fff.eq.3) write(49,6013)
        if (fff.eq.4) write(49,6014)
6011    format('Exploration Wells Drilled')
6012    format(///,'Development Wells Drilled')
6013    format(///,'Total Wells Drilled')
6014    format(///,'Total Operating Wells')
        write(49,2331)(tme(t),t=1,ntme)

        do s=1,nsrg
        if (fff.eq.1) write(49,5172) srgnme(s),(spsenw(t,s),t=1,ntme)
        if (fff.eq.2) write(49,5172) srgnme(s),(spsdwp(t,s)+spsdws(t,s),
        t=1,ntme)
        if (fff.eq.3) write(49,5172) srgnme(s),(spsenw(t,s)+spsdwp(t,s)+
        spsdws(t,s),t=1,ntme)
        if (fff.eq.4) write(49,5172) srgnme(s),(spsnpw(t,s),t=1,ntme)
        enddo
        if (fff.eq.1) write(49,5173) (spsenwus(t),t=1,ntme)
        if (fff.eq.2) write(49,5173) (spsdwpus(t)+spsdwsus(t),
        t=1,ntme)
        if (fff.eq.3) write(49,5173) (spsenwus(t)+spsdwpus(t)+
        spsdwsus(t),t=1,ntme)
        if (fff.eq.4) write(49,5173) (spsnpwus(t),t=1,ntme)

        do ir=1,(2*mxrsty)
        mir=ir
        if(mir.gt.mxrsty)mir=ir-mxrsty
        if(ir.eq.mxrsty+1) then
        if (fff.eq.1) write(49,5174) (spsenwcn(t),t=1,ntme)
        if (fff.eq.2) write(49,5174) (spsdwpcn(t)+spsdwscn(t),
        t=1,ntme)
        if (fff.eq.3) write(49,5174) (spsenwcn(t)+spsdwpcn(t)+
        spsdwscn(t),t=1,ntme)
        if (fff.eq.4) write(49,5174) (spsnpwcn(t),t=1,ntme)
        endif
        if (fff.eq.1) write(49,5172) rstynm(mir),
        (spsenwir(t,ir),t=1,ntme)
        if (fff.eq.2) write(49,5172) rstynm(mir),
        (spsdwpir(t,ir)+spsdwsir(t,ir),t=1,ntme)
        if (fff.eq.3) write(49,5172) rstynm(mir),
        (spsenwir(t,ir)+spsdwpir(t,ir)+spsdwsir(t,ir),t=1,ntme)
        if (fff.eq.4) write(49,5172) rstynm(mir),
        (spsnpwir(t,ir),t=1,ntme)
        enddo
6001    continue
```

Note: 'Suppsumm.out', 'wellsumm.out', 'decision.out', 'supply.ext', etc. are closed.

```
        close(48)
        close(49)
        close(43)
        close(40)
        close(51)
        close(97)
```

Note: The supply summary reports are written.
'nsrg' represents the total number of supply regions (23 in all).
'nrst' represents the total number of resource types (7 in all)

```
      nn=1+2+nsrg+nrst+nrst

      do 6100 ix1=1,nn
      if(ix1.eq.1) then
      ip1=1
      ip2=nply
      else
      ip1=nply+1
      ip2=nply+1
      p=ip1
```

Note: Various variables are initialized.

```
      splirptot=0.0
      splirstot=0.0
      do 5192 y=1,nyr
      splirs(p)=0.0
      splrsp(y,p)=0.0
      splrss(y,p)=0.0
      splprd(y,p)=0.0
      splenw(y,p)=0.0
      spledc(y,p)=0.0
      sploec(y,p)=0.0
      spldws(y,p)=0.0
      spldcs(y,p)=0.0
      splndc(y,p)=0.0
      splfom(y,p)=0.0
      splnpw(y,p)=0.0
      splvom(y,p)=0.0
      spldwp(y,p)=0.0
      splrfc(y,p)=0.0
      spldss(y)=0.0
      spxnpw(y,p)=0.0
      spxprd(y,p)=0.0
      splwlr(y)=0.0
      do 5191 f=1,mxnfsz
      plybcf(1,f,y)=0.0
      plybcf(2,f,y)=0.0
      plybcf(3,f,y)=0.0
      plysvx(f,y)=0.0
      if(y.eq.nyr) then
      plysvx(f,y+1)=0.0
      endif
5191  continue
5192  continue
      do 5196 ip3=1,nply
      s=plyrga(ip3)
```

Note: Pointers for reporting are assigned as follows.

'icde' is 0 for US
icde' is 1 for Alberta, British Columbia, MacKenzie Delta, Sable Island
'icde' is 2 for Mexico-Supply

```
icde=0
if((srgnme(s).eq.'Alberta            ').or.
(srgnme(s).eq.'British Columbia    ').or.
(srgnme(s).eq.'MacKenzie Delta     ').or.
(srgnme(s).eq.'Sable Island        ')) then
icde=1
endif
if (srgnme(s).eq.'Mexico-Supply       ') then
icde=2
endif

if((ix1.eq.2).and.(icde.eq.1)) go to 5196
if((ix1.eq.2).and.(icde.eq.2)) go to 5196
if((ix1.eq.3).and.(icde.ne.1)) go to 5196
if(((ix1.gt.3).and.(ix1.le.(3+nsrg))).and.(s.ne.ix1-3))
go to 5196
if((ix1.gt.(3+nsrg)).and.(ix1.le.(3+nsrg+nrst)).and.
(icde.ne.0)) go to 5196
if((ix1.gt.(3+nsrg)).and.(ix1.le.(3+nsrg+nrst)).and.
(prtyp(ip3).ne.(ix1-3-nsrg))) go to 5196
if((ix1.gt.(3+nsrg+nrst)).and.(icde.ne.1)) go to 5196
if((ix1.gt.(3+nsrg+nrst)).and.(prtyp(ip3).ne.
(ix1-3-nsrg-nrst))) go to 5196
```

Note: All the variables are assigned values and are cumulative functions.

```
splirs(p)=splirs(p)+splirs(ip3)
do 5195 y=1,nyr
splrsp(y,p)=splrsp(y,p)+splrsp(y,ip3)
splrss(y,p)=splrss(y,p)+splrss(y,ip3)
splirptot=splirptot+splrsp(y,ip3)/1000.
splirstot=splirstot+splrss(y,ip3)/1000.
splprd(y,p)=splprd(y,p)+splprd(y,ip3)
splenw(y,p)=splenw(y,p)+splenw(y,ip3)
spledc(y,p)=spledc(y,p)+spledc(y,ip3)
sploec(y,p)=sploec(y,p)+sploec(y,ip3)
spldws(y,p)=spldws(y,p)+spldws(y,ip3)
spldcs(y,p)=spldcs(y,p)+spldcs(y,ip3)
splndc(y,p)=splndc(y,p)+splndc(y,ip3)
splfom(y,p)=splfom(y,p)+splfom(y,ip3)
splnpw(y,p)=splnpw(y,p)+splnpw(y,ip3)
splvom(y,p)=splvom(y,p)+splvom(y,ip3)
spldwp(y,p)=spldwp(y,p)+spldwp(y,ip3)
splrfc(y,p)=splrfc(y,p)+splrfc(y,ip3)
spldss(y)=spldss(y)+plydsc(wi,ip3)*(spldws(y,ip3)
+spldwp(y,ip3))
```

```
        spxnpw(y,p)=spxnpw(y,p)+spxnpw(y,ip3)
        spxprd(y,p)=spxprd(y,p)+spxprd(y,ip3)
        splwlr(y)=splwlr(y)+supprc(y,s)*splprd(y,ip3)
        do 5194 f=1,mxnfsz
        plybcf(1,f,y)=plybcf(1,f,y)+
        plysve(1,f,ip3,y)*ogip(f,ip3)/1000000.0
        plybcf(2,f,y)=plybcf(2,f,y)+
        plysve(2,f,ip3,y)/1000000.0
        plybcf(3,f,y)=plybcf(3,f,y)+
        plysve(3,f,ip3,y)/1000000.0
        plysvx(f,y)=plysvx(f,y)+plysve(1,f,ip3,y)
        if(y.eq.nyr) then
        plysvx(f,y+1)=plysvx(f,y+1)+plynfl(f,ip3)
        endif
5194    continue
5195    continue
5196    continue
        endif
        do 6000 p=ip1,ip2
        icdx=1
        if(ix1.eq.1) then
        s=plyrga(p)
        icdx=iwrt_play
        icdxfp=0
        do 5197 f=1,mxnfsz
        if(plysve(1,f,p,1).gt.0.0.or.
        plysve(2,f,p,1).gt.0.0) icdxfp=1
5197    continue
```

Note: The output is printed in the ‘prodsumm.out’ file.

```
        if (icdx.eq.1.and.icdxfp.eq.0) icdx=0
        if(icdx.ne.0) write(44,5201) srgnme(s),plynme(p),rstynm(rst(p))
5201    format(' ',t50,' Supply Summary Report',/,' Region: ',a20/,
        ' Play:   ',a20,' Res:  ',a20)
        elseif(ix1.eq.2) then
        write(44,5201) usname,allnme
        s=14
        elseif(ix1.eq.3) then
        write(44,5201) cnname,allnme
        s=18
        elseif(ix1.le.(3+nsrg)) then
        write(44,5201) srgnme(ix1-3),allnme
        s=ix1-3
        elseif(ix1.le.(3+nsrg+nrst)) then
        write(44,5207) usname,rstynm(ix1-(3+nsrg))
        s=14
        else
        write(44,5207) cnname,rstynm(ix1-(3+nsrg+nrst))
        s=18
5207    format(' ',t50,' Supply Summary Report',/,' Region: ',a20/,
        ' Type:   ',a20)
5208    format(' ',t50,' Reserve Summary Report',/,' Region: ',a20/,
        ' Type:   ',a20)
        endif
        if(icdx.ne.0) write(44,5202) (y,y=tmex,tmex+nyr-1)
5202    format(t20,40(2x,i4,2x))
        if(icdx.ne.0) write(44,5203)
```

```
5203      format(' ')
```

Note: All the variables are assigned values under specific conditions.

```
          splboy(1)=splirs(p)
          do 5204 f=1,mxnfsz
          ogip(f,p)=0.0
5204      continue
          do 5206 g=1,nnfl
          ii=nnflb(g)/mxnfsz
          if(ii.eq.p) then
          prtyp(p)=ndrsty(g)
          f=nnflb(g)-p*mxnfsz+1
          do 5205 d=1,mxngr
          ogip(f,p)=ogip(f,p)+ndogip(d,g)
5205      continue
          endif
```

Note: Field size class and play is assigned.

```
5206      continue
          do 5230 y=1,nyr
```

Note: ‘spleoy’ represents the end of year resources.

```
          spleoy(y)=splboy(y)+splrsp(y,p)+splrss(y,p)-splprd(y,p)
          splrsa(y)=splrsp(y,p)+splrss(y,p)
          if(y.lt.nyr) then
          splboy(y+1)=spleoy(y)
          val=0.0
          do 5210 f=1,mxnfsz
          if(ix1.eq.1) then
          if(y.eq.1) then
          plysvx(f,y)=plysve(1,f,p,y)
          endif
          plysvx(f,y+1)=plysve(1,f,p,y+1)
          endif
          val=val+plysvx(f,y)-plysvx(f,y+1)
5210      continue
          else
          val=0.0
          do 5220 f=1,mxnfsz
          if(ix1.eq.1) plysvx(f,y+1)=plynfl(f,p)
          val=val+plysvx(f,y)-plysvx(f,y+1)
5220      continue
          endif
          if(splenw(y,p).gt.0.0) then
          spless(y)=val/splenw(y,p)*100.0
          spless(y)=spless(y)/3.0
          else
          spless(y)=0.0
          endif
          if(splenw(y,p).gt.0.0) then
          spledx(y)=spledc(y,p)/splenw(y,p)
          splenx(y)=sploec(y,p)/splenw(y,p)
```

```
else
spledx(y)=0.0
splenx(y)=0.0
endif
spltcx(y)=spledx(y)+splenx(y)
spltec(y)=spledc(y,p)+sploec(y,p)
spldwx(y)=spldwp(y,p)+spldws(y,p)
if(spldwx(y).gt.0.0) then
splddx(y)=spldcs(y,p)/spldwx(y)
spldnx(y)=splndc(y,p)/spldwx(y)
else
splddx(y)=0.0
spldnx(y)=0.0
endif
spldtx(y)=splddx(y)+spldnx(y)
spltdc(y)=spldcs(y,p)+splndc(y,p)
if(splnpw(y,p).gt.0.0) then
splfow(y)=splfom(y,p)/splnpw(y,p)
else
splfow(y)=0.0
endif
if(splprd(y,p).gt.0.0) then
splvow(y)=splvom(y,p)/splprd(y,p)
spltop(y)=(splfom(y,p)+splvom(y,p))/splprd(y,p)
else
splvow(y)=0.0
spltop(y)=0.0
endif
spltom(y)=splfom(y,p)+splvom(y,p)
spltcs(y)=spltom(y)+spltdc(y)+spltec(y)
spltwl(y)=splenw(y,p)+spldwp(y,p)+spldws(y,p)
if(spldwp(y,p).gt.0.0) then
splrwp(y)=splrsp(y,p)/spldwp(y,p)
else
splrwp(y)=0.0
endif
if((spldws(y,p)+splrfc(y,p)).gt.0.0) then
splrws(y)=splrss(y,p)/(spldws(y,p)+splrfc(y,p))
else
splrws(y)=0.0
endif
spyprd(y)=splprd(y,p)-spxprd(y,p)
spynpw(y)=splnpw(y,p)-spxnpw(y,p)
if((spldwp(y,p)+spldws(y,p)+splrfc(y,p)).gt.0.0) then
splrwa(y)=splrsa(y)/(spldwp(y,p)+spldws(y,p)+splrfc(y,p))
else
splrwa(y)=0.0
endif
if(spxnpw(y,p).gt.0.0) then
spxrte(y)=spxprd(y,p)/spxnpw(y,p)*1000000.0/365.
else
spxrte(y)=0.0
endif
if(splnpw(y,p).gt.0.0) then
splrte(y)=splprd(y,p)/splnpw(y,p)*1000000.0/365.
else
splrte(y)=0.0
endif
if(spynpw(y).gt.0.0) then
spyrte(y)=spyprd(y)/spynpw(y)*1000000.0/365.
else
spyrte(y)=0.0
```

```
        endif
        if(splprd(y,p).gt.0.0) then
        splrp(y)=(splboy(y)+splrsa(y))/splprd(y,p)
        else
        splrp(y)=0.0
        endif
        if(ix1.eq.1) then
        splwlr(y)=supprc(y,s)*splprd(y,p)
        endif
        spltud(y)=0.0
        spltbn(y)=0.0
        spltb_rg(y)=0.0
        do 5225 f=1,mxnfsz
        if(ix1.eq.1) then
```

Note: Variables are being converted from MMCF to TCF.

```
        plybcf(1,f,y)=plysve(1,f,p,y)*ogip(f,p)/1000000.0
        plybcf(2,f,y)=plysve(2,f,p,y)/1000000.0
        plybcf(3,f,y)=plysve(3,f,p,y)/1000000.0

        endif
        spltud(y)=spltud(y)+plybcf(1,f,y)
        spltbn(y)=spltbn(y)+plybcf(2,f,y)
        spltb_rg(y)=spltb_rg(y)+plybcf(3,f,y)
5225    continue
        if(ix1.eq.1) then
        spldss(y)=plydsc(wi,p)*100.0
        else
        if((spldwp(y,p)+spldws(y,p)).gt.0.0) then
        spldss(y)=spldss(y)/(spldwp(y,p)+spldws(y,p))*100.0
        else
        spldss(y)=0.0
        endif
        endif
5230    continue
```

Note: The output is printed in the reserves and production summaries.

```
        if(icdx.ne.0) then
        write(44,5231)
5231    format(' Reserves/Production Summary (BCF): ')
        write(44,5232) (splboy(y),y=1,nyr)
5232    format('   BOY Reserves: ',t20,40(1x,f7.0))
        write(44,5233) (splrsa(y),y=1,nyr)
5233    format('   Res. Adds: ',t20,40(1x,f7.0))
        write(44,5234) (splprd(y,p),y=1,nyr)
5234    format('   -Production: ',t20,40(1x,f7.0))
        write(44,5235) (spleoy(y),y=1,nyr)
5235    format('   EOY Reserves: ',t20,40(1x,f7.0))

        write(44,5271)
5271    format(/' Drilling Summary (wells): ')
        write(44,5272) (splenw(y,p),y=1,nyr)
5272    format('   Exploration: ',t20,40(1x,f7.0))
        write(44,5273) (spldwp(y,p),y=1,nyr)
```

```
5273      format('   Devl-Primary: ',t20,40(1x,f7.0))
          write(44,5274) (spldws(y,p),y=1,nyr)
5274      format('   Devl-Infill: ',t20,40(1x,f7.0))
          write(44,5275) (spltwl(y),y=1,nyr)
5275      format('   Total: ',t20,40(1x,f7.0))
          write(44,5276) (splrfc(y,p),y=1,nyr)
5276      format('   Recompletes: ',t20,40(1x,f7.0))
          write(44,5277) (spless(y),y=1,nyr)
5277      format(/'   Exp. Success Rate (%): ',t20,40(1x,f7.0))
          write(44,5278) (spldss(y),y=1,nyr)
5278      format('   Devl Success Rate (%): ',t20,40(1x,f7.0))

          write(44,5281)
5281      format(/' Reserve Addition Summary (Bcf): ')
          write(44,5282) (splrsp(y,p),y=1,nyr)
5282      format('   Primary: ',t20,40(1x,f7.0))
          write(44,5283) (splrss(y,p),y=1,nyr)
5283      format('   Secondary: ',t20,40(1x,f7.0))
          write(44,5284) (splrsa(y),y=1,nyr)
5284      format('   Total: ',t20,40(1x,f7.0))
          write(44,5285)
5285      format(/' Reserve Addition Summary (Bcf/Well): ')
          write(44,5286) (splrwp(y),y=1,nyr)
5286      format('   Primary: ',t20,40(1x,f7.2))
          write(44,5287) (splrws(y),y=1,nyr)
5287      format('   Secondary: ',t20,40(1x,f7.2))
          write(44,5288) (splrwa(y),y=1,nyr)
5288      format('   Total: ',t20,40(1x,f7.2))

          write(44,5291) tmex
5291      format(/' Production Summary - Pre ',i4,1x,'Fields:')
          write(44,5292) (spxprd(y,p),y=1,nyr)
5292      format('   Prod (Bcf): ',t20,40(1x,f7.0))
          write(44,5293) (spxnpw(y,p),y=1,nyr)
5293      format('   Number Wells: ',t20,40(1x,f7.0))
          write(44,5294) (spxrte(y),y=1,nyr)
5294      format('   Avg Rate (mcf/day/well): ',t20,40(1x,f7.1))
          write(44,5295) tmex
5295      format(/' Production Summary - ',i4,1x,'and Beyond Fields:')
          write(44,5292) (spyprd(y),y=1,nyr)
          write(44,5293) (spynpw(y),y=1,nyr)
          write(44,5294) (spyrte(y),y=1,nyr)
          write(44,5296)
5296      format(/' Production Summary - All Fields:')
          write(44,5292) (splprd(y,p),y=1,nyr)
          write(44,5293) (splnpw(y,p),y=1,nyr)
          write(44,5294) (splrte(y),y=1,nyr)
          write(44,5297) (splrp(y),y=1,nyr)
5297      format( '   R/P ratio: ',t20,40(1x,f7.2))
          write(44,5298) (supprc(y,s),y=1,nyr)
5298      format(/'    WH Price ($/mcf): ',t20,40(1x,f7.2))
          write(44,5299) (splwlr(y),y=1,nyr)
5299      format('    Revenues ($MM): ',t20,40(1x,f7.0))

          write(44,5301)
5301      format(/' Origial Gas in Place Summary (Tcf):')
          write(44,5302)
5302      format(' '/' Undiscovered Resource (OGIP) at BOY')
          do 5310 f=mxnfsz,1,-1
          ii=mxnfsz-f+1+4
          write(44,5303) ii,(plybcf(1,f,y),y=1,nyr)
5303      format('  Sz Class: ',i2,t20,40(1x,f7.3))
```

```
5310     continue
         write(44,5311) (spltud(y),y=1,nyr)
5311     format( '  Total: ',t20,40(1x,f7.2))
         write(44,5312)
5312     format(' '/' Banked Resource (OGIP) at BOY')
         do 5320 f=mxnfsz,1,-1
         ii=mxnfsz+1-f+4
         write(44,5303) ii,(plybcf(2,f,y),y=1,nyr)
5320     continue
         write(44,5311) (spltbn(y),y=1,nyr)

         write(44,5315)
5315     format(' '/' Reserve Growth Resource (OGIP) at BOY')
         do 5321 f=mxnfsz,1,-1
         ii=mxnfsz+1-f+4
         write(44,5303) ii,(plybcf(3,f,y),y=1,nyr)
5321     continue

         write(44,5311) (spltb_rg(y),y=1,nyr)

         IF (spltud(1).gt.0.0.or.splboy(2).gt.0.0) THEN
         if(icdx.ne.0.and.ix1.eq.1) write(47,5208) srgnme(s),plynme(p)
         if(ix1.eq.2) then
         write(47,5208) usname,allnme
         elseif(ix1.eq.3) then
         write(47,5208) cnname,allnme
         elseif(ix1.le.(3+nsrg)) then
         write(47,5208) srgnme(ix1-3),allnme
         else
         write(47,5208) usname,rstynm(ix1-(3+nsrg))
         endif
         write(47,5398)
         spltud(1)+spltbn(1),spltud(1),spltbn(1),
         spltud(1)-spltud(nyr),spltbn(1) + spltud(1) - spltud(nyr),
         spltud(1)-spltud(nyr)
         +spltbn(1)-spltbn(nyr)
         write(47,5499)
         splboy(1)/1000.,splirptot,splirstot,splboy(1)/1000.+splirptot+
         splirstot
         endif
         ENDIF ! skip writing if no resource

5398     format(/,' Total Undiscovered and Undeveloped Resource (TCF): ',
         t49,1x,f10.3,/,
         ' Total Undiscovered Resource (TCF): ',t49,1x,f10.3,/,
         ' Discovered Undeveloped Resource (TCF): ',t49,1x,f10.3,/,
         ' Total Discovered (TCF): ',t49,1x,f10.3,/,
         ' Total Available for Development (TCF): ',t49,1x,f10.3,/,
         ' Total Developed (TCF): ',t49,1x,f10.3,/)
5499     format(//,' Initial Reserves (TCF): ',t49,1x,f10.3,/,
         ' Reserve Additions (Prim.) (TCF):',t49,1x,f10.3,/,
         ' Reserve Additions (Sec.) (TCF):',t49,1x,f10.3,/,
         ' Total Reserves/Prod (TCF): ',t49,1x,f10.3,/)

6000     continue
6100     continue
```

<u>Step 3</u>: **The subroutine ends.**

```
return
end
```

Table of Contents

SUBROUTINE ENV_WRTE

CALLS: ILOOK (Compares two variables from different sources)
NONZEROS (Looks at only nonzero values)
PROCESS (Calculates minimum costs for gas processing/treatment.)
PACKVAL (Assigns values to certain local variables)

READS: ‘sequen.dat’ (Specifies the files to be included in the construction of the environmental and processing files.)
‘statereg.env’ (Contains state/regional environmental costs.)

CREATES: ‘env_stat.out’ (Contains static cost data.)
‘env_proc.out’ (Contains processing cost data.)
‘env3.out’ (Contains environmental data.)

MAIN THEME: The environmental costs are read in by state and then written by GSAMID. In addition, the gas processing cost file is also created.

Step 1: **The input file 'sequen.dat' is opened (from the ‘news’ subdirectory).**

Note: This file specifies the files to be included in the construction of the environmental and processing files. The sequence of ‘*.env’ files in ‘sequence.dat’ should be exactly the same as in ‘spec.dtu’ and ‘spec.dtd’ files.

```
open(unit=15,file='..\explprod\news\sequen.dat',status='old')
```

Step 2: **The input file 'statereg.env' is opened. This file contains state/regional environmental costs.**

```
open(unit=20,file='statereg.env')
```

Step 3: **Three output files are opened.**

Note: 'env_stat.out' contains static
data, 'env_proc.out' contains processing cost data, and
'env3.out' contains environmental data. The common index
for each of these files is the GSAMID.

```
open (unit=51,file='env_stat.out')
open (unit=52,file='env_proc.out')
open (unit=53,file='env3.out')

write(6,*) '================================================='
write(6,*) ' The Sequence of *.ENV files in SEQUEN.DAT should'
write(6,*) ' be exactly same as in SPEC.DTU & SPEC.DTD Files'
write(6,*) '================================================='
```

Step 4: **Various variables are initialized.**

Note: The reservoir number (resnum) is initially set to zero. The maximum number of state/region entries in the file statereg.env (ntotal) is set to 1.

```
resnum = 0
ntotal = 1
```

Step 5: **The code begins the process of looking up state/regional environmental costs.**

Note: The costs are read from the state/regional level environmental costs file. In the following read statements, the header lines are read. There are five header lines at the start of the environmental output file (such as '98DOE.ENV' in the ..\GSAM\EXPLPROD\ENV sub-directory.)

```
read(20,*)
read(20,*)
read(20,*)
read(20,*)
read(20,*)
```

Step 6: **Environmental variables are read.**

Note: The following environmental variables are read by supply region (ireg):

existing well tangible cost (eetcap) (thousands of $ per well)
existing well intangible cost (eeicap) (thousands of $ per well)
existing well O&M cost (eeoam) (thousands of $ per well)
new well tangible cost (nntcap) (thousands of $ per well)

new well intangible cost (nnicap) (thousands of $ per well)
new well O&M cost (nnoam) (thousands of $ per well)

```
10      read(20,*,end=30)ireg(ntotal),eetcap(ntotal),
        eeicap(ntotal),eeoam(ntotal),
        nntcap(ntotal),nnicap(ntotal),
        nnoam(ntotal)
```

Step 7: **Variables are initialized.**

Note: The following environmental costs are currently set to zero.
environmental drilling costs per foot (eenvdcf) ($/ft)
environmental gas costs (eenvgc) ($/MCF)
environmental water costs (eenvwc) ($/barrel)

```
        eenvdcf(ntotal)= 0.0
        eenvgc(ntotal) = 0.0
        eenvwc(ntotal) = 0.0

        ntotal   = ntotal + 1
        go to 10

30      ntotal = ntotal - 1
```

Step 8: **Variables are initialized.**

Note: The following values initialized first. Royalty rate (this gets over-written later on by the read statement in step 11) to 12.5% (royrate), lease bonus ($/Acre) (lsb) to zero and condensate yield (bbl of NGL/MMCF dry gas) (watyld) to zero.

```
        royrate = 0.125   ! default number
        lsb     = 0.0     ! default number
        watyld  = 0.0     ! default number
```

Step 9: **The input file 'sequen.dat' is opened and the corresponding file name (such as 98DOE.ENV) is read.**

```
98      read(15,'(a25)',end=99) file_in
        if (file_in.eq.' ')then
        stop
        endif
```

Step 10: **The file read in step 9 (such as 98DOE.ENV, specified in 'sequence.dat') is opened.**

```
        open(unit=10,file=file_in)

        write(6,'(1x,a,a25)') 'Analyzing = ', file_in
```

Step 11: **The code begins the process of reading variables at the reservoir level.**

Note: Variables read include
GSAMID, state, depth, area, royalty rate (fraction), fraction of the reservoir which is on federal lands, CO2 content (fraction), N2 content (fraction), H2S (fraction), and condensate yield (bbl NGL/MMCF dry gas). These entries have come from the reservoir performance module.

```
40      read(10,9773,end=60) gsamid,state,depth,area,royrate,
        frac_fed,co2,n2,h2s,condyld
9773    format(a11,i5,1x,f7.0,1x,f10.0,1x,f7.3,1x,f7.3,1x,f7.5,1x,
        f7.5,f7.5,1x,f11.3)

        regname=gsamid(1:2)
        ngl=condyld
```

Step 12: **Here the code checks to ensure the right match between reservoir-level and state-level values using the ilook subroutine.**

```
        ichk = 0
        call ilook(state,ireg,ntotal,ichk)
        if (ichk.eq.0) then
        write(*,*)' state/region mismatch for ',state
        etcap = 0.
        eicap = 0.
        eoam  = 0.
        ntcap = 0.
        nicap = 0.
        noam  = 0.
        envdcf= 0.
        envgc = 0.
        envwc = 0.
        goto 150
        end if
```

Step 13: **The variables are assigned for every reservoir based on the 'state match' counter (ichk).**

```
        etcap = eetcap(ichk)
        eicap = eeicap(ichk)
        eoam  = eeoam(ichk)
        ntcap = nntcap(ichk)
        nicap = nnicap(ichk)
        noam  = nnoam(ichk)
        envdcf= eenvdcf(ichk)
        envgc = eenvgc(ichk)
        envwc = eenvwc(ichk)
```

Step 14: **The nonzero values are packed.**

Note: The nonzero values are packed into the variables 'outnum' and 'outval'. These are printed in the environmental files (in step 17) that get passed over to the E&P module.

```
150     varptr  = 0
        call nonzeros(etcap,eicap,eoam,ntcap,nicap,
        noam,envdcf,envgc,envwc,
        varptr,outval,outnum)
```

Step 15: **The code begins the procedure of calculating processing costs.**

```
        resnum = resnum + 1
```

Step 16: **The subroutine 'process.for' is called.**

Note: Process.for calculates processing costs for CO2, N2, H2S and NGL and provides information by GSAMID in $/MCF.

```
        call process(n2,co2,h2s,ngl,totnet,resnum,gsamid)
```

Step 17: **The data is written in the three output files, 'env_stat.out', 'env_proc.out', 'env3.out'.**

```
        write(51,905) gsamid,state,depth,area,royrate,frac_fed,co2,
        n2,h2s,lsb,condyld,watyld
905     format(a11,1x,i4,1x,f12.4,1x,f14.4,1x,f12.4,1x,f7.3,1x,f12.4,
        1x,f12.4,1x,f12.4,1x,f12.4,1x,f12.4,1x,f12.4)

        write(52,907) gsamid,totnet
        write(53,910) gsamid,varptr
        do ii=1,varptr
        write(53,915)outnum(ii),outval(ii)
        enddo

        go to 40

60      close(10)
        goto 98

99      stop

907     format(a11,1x,f12.4)
910     format(a11,1x,i2)
915     format(i2,1x,f12.4)
```

Step 18: **The subroutine ends.**

```
        End
```

Step 19: **The packval routine assigns variable numbers (outnum) and a value (outval) to each variable according to the variable pointer (varptr).**

```
        Subroutine packval(varnum,varval,outnum,outval,varptr)
        integer varptr,varnum,outnum(9)
        real  varval,outval(9)

        varptr = varptr + 1
        outnum(varptr) = varnum
        outval(varptr) = varval

        return
        end
```

Step 20: **The ‘nonzero’ routine assigns to ‘varval’ (subsequently used in ‘packval’) the nonzero value of all the variables in question (listed in the comment statement below).**

Note: This routine in conjunction with ‘packval’ is used to filter out those variables with a value of zero. This is done to speed up the analysis.

```
        Subroutine nonzeros(etcap,eicap,eoam,ntcap,nicap,
1       noam,envdcf,envgc,envwc,
2        varptr,outval,outnum)

        integer varptr,varnum,outnum(9)
        real varval,outval(9)
        real etcap,eicap,eoam
        real ntcap,nicap,noam,envdcf,envgc,envwc


        if (etcap.ne.0.0) then
        varnum = 1
        varval = etcap
        call packval(varnum,varval,outnum,outval,varptr)
        end if

        if (eicap.ne.0.0) then
        varnum = 2
        varval = eicap
        call packval(varnum,varval,outnum,outval,varptr)
        end if

        if (eoam.ne.0.0) then
        varnum = 3
        varval = eoam
        call packval(varnum,varval,outnum,outval,varptr)
        end if

        if (ntcap.ne.0.0) then
        varnum = 4
        varval = ntcap
        call packval(varnum,varval,outnum,outval,varptr)
        end if

        if (nicap.ne.0.0) then
        varnum = 5
        varval = nicap
        call packval(varnum,varval,outnum,outval,varptr)
        end if

        if (noam.ne.0.0) then
        varnum = 6
        varval = noam
        call packval(varnum,varval,outnum,outval,varptr)
        end if

        if (envdcf.ne.0.0) then
        varnum = 7
        varval = envdcf
        call packval(varnum,varval,outnum,outval,varptr)
        end if

        if (envgc.ne.0.0) then
        varnum = 8
        varval = envgc
        call packval(varnum,varval,outnum,outval,varptr)
        end if

        if (envwc.ne.0.0) then
        varnum = 9
        varval = envwc
        call packval(varnum,varval,outnum,outval,varptr)
        end if


        return
        end
```

SUBROUTINE PROCESS.FOR

CALLED BY: ENV_WRTE (This subroutine writes the environmental costs.)

CALLS: This routine calls a series of subroutines which have been documented below.

READS: 'pro_reg.spc' (Contains the capacity of various plants by region.)
'pro_dat.spc' (Contains limits permissible for impurities and plants.)

CREATES: 'process.out' (Contains the Gas Processing Output Report)
'process2.out' (Contains the Gas Processing Output Report)
'process3.out' (Contains the Gas Processing Output Report)

MAIN THEME: This program computes costs in $/MCF for the gas processing procedure. These costs also include revenue from associated byproducts. The basis of calculations for CO2, H2S and N2 removal and credit for associated by products is the following reports:
(1) "Business Characteristics of the Natural Gas Conditioning Industry"', Topical Report (May 1993), Prepared by : C.C. Tannehill and C. Galvin, Purvin & Gertz, Inc.,Prepared for: The M. W. Kellogg Company Gas Research Institute, Task 40,GRI Contract No. 5088-221-1753,
(2) "Natural Gas Processing Costs: Cost Equations for Various Technologies," ICF Resources, Inc.

Step 1: **The output file, 'process.out', is opened.**

Note: This file may contain information about consistency checks.

```
open(unit=14,file='process.out')
```

Step 2: **The second output , 'process2.out', file is opened.**

Note: This file may contain information about consistency checks.

```
open(unit=22,file='process2.out')
```

Step 3: **The regional inputs from proc_reg.spc are read in, each record refers to a specific region.**

```
      open(unit=41,file='pro_reg.spc')
```

Step 4: The counter for region number (regnum) is initialized.

```
      regnum = 0
5     regnum = regnum + 1
```

Step 5: A number of variables are read in from the ‘pro_reg.spc’ file.

Note: The following are read in: region, inlet capacity in MMCFD and gas thruput in MMCFD.
'capsu()' is the capacity (long tons/day) for Claus Sulfur Recovery and Direct Conversion of H2S to Sulfur (Chelated Iron Process), 'capst()' is the capacity (MMCFD) for NGL Straight Refrigeration, 'capcr()' is the capacity (MMCFD) for NGL Cryogenic Expander Plant, ‘thru()’ is the ngl thruput (used for all impurities) and 'capte()' is the capacity (MMCFD) for all other plant types based on the averages for Texas (assumed to be representative of the U.S.).

```
      read(41,900,end=10)region(regnum),capst(regnum),capcr(regnum),
      capte(regnum),capsu(regnum),thru(regnum),op_fac(regnum)
```

Step 6: 'Nglnumber' is a fraction calculated based on ngl (bbl/MMCF) value for a reservoir.

Note: The ngl value is a database element in the *.gsm file.

```
      nglnumber = (ngl*1512.0)/(10**6 + ngl*1512.0)
      denom = 1.0 - n2 - co2 - h2s - nglnumber

      if (denom.gt.0.0) then
      capst(regnum) = capst(regnum)/denom
      capcr(regnum) = capcr(regnum)/denom
      capte(regnum) = capte(regnum)/denom
      capsu(regnum) = capsu(regnum)/denom
      else
      write(6,*) 'The Following GSAMID has Very High Impurities'
      write(6,*) gsamid,denom
      endif

      call check1(regnum,capst(regnum),capcr(regnum),capte(regnum),
      capsu(regnum),thru(regnum),op_fac(regnum))
      go to 5
```

```
10      regnum = regnum -1
```

Step 7: **The discount rate (dis_rate), plant life (plt_lfe) and the impurity limits (N2_lim, CO2_lim, H2O_lim, H2S_lim, ngl_lim) are read in as fractions (in years) from the 'pro_dat.spc' file.**

Note: The limits define the maximum levels permissible in the gas stream.

```
        open(unit=43,file='pro_dat.spc')
        read(43,*)  dis_rate,plt_lfe,n2_lim,co2_lim,h2o_lim,
        h2s_lim,ngl_lim

        call check3(dis_rate,plt_lfe,n2_lim,co2_lim,h2o_lim,
        h2s_lim,ngl_lim)
```

Step 8: **The two digit GSAM region name is stored from the GSAMID.**

```
        regname=gsamid(1:2)
        h2o= 0.00
```

Note: For the corresponding region (in which the reservoir is located) entries for inlet capacities, gas thruput, operating cost factors etc. are assigned. These entries are specified in the 'pro_reg.spc' file.

```
        do i=1,regnum
        if (regname.eq.region(i)) then
        capstrt  = capst(i)
        capcryo  = capcr(i)
        captexas = capte(i)
        capsulf  = capsu(i)
        thruput  = thru(i)
        op_factor = op_fac(i)
        go to 20
        end if
        enddo
```

Note: If there is a mismatch in the region name, the program halts.

```
        write(*,*)'Region name mismatch'
        write(*,*)'regname=',regname
        write(*,*)'region=',(region(i),i=1,regnum)
        write(*,*)'Program halted'
        stop

20      call check2(resnum,n2,co2,h2o,h2s,ngl)
```

Step 9: **The total acid gas concentration (in fraction) is calculated by adding H2S and CO2 concentrations.**

```
acid = h2s+co2
```

Step 10: **The total gas processing cost is calculated according to the equation gas processing cost = capital cost + operating cost - byproduct revenues.**

Note: First the capital cost is calculated, this is followed by the calculation of the operating cost, which is followed by the calculation of the byproduct revenues.

Note: Here capital costs are calculated (cap_cost subroutine).

```
call cap_cost(capstrt,capcryo,captexas,capsulf,acid,
n2,n2_lim,
co2,co2_lim,h2s,h2s_lim,
h2o,h2o_lim,
ngl,ngl_lim,
dis_rate,plt_lfe,op_factor,
thruput,
clccost,diccost,stccost,crccost,
n2_ccst,co2_ccst,h2o_ccst,ngl_ccst)
```

Note: Here operating costs ($/MCF) are calculated (op_cost subroutine).

```
call op_cost(capstrt,capcryo,captexas,capsulf,thruput,
acid,
n2,n2_lim,
co2,co2_lim,h2s,h2s_lim,
h2o,h2o_lim,
ngl,ngl_lim,
diocost,stocost,crocost,
n2_ocst,co2_ocst,ngl_ocst,h2o_ocst)
```

Note: Here the byproduct revenues ($/MCF) are calculated (by_rev subroutine).

```
call by_rev(byrevs,h2s,co2,ngl)

imp(1) ='n2'
imp(2) ='co2+h2s'
imp(3) ='h2o'
imp(4) ='ngl'
imp(5) ='TOTAL'
```

Note: Total gas processing cost ($/MCF) for N2 is calculated by adding the operating cost and capital cost for removing nitrogen from the gas stream.

```
n2_cst = n2_ocst/denom+n2_ccst
```

Note: Total gas processing cost ($/MCF) for CO2 is calculated by adding the operating cost and capital cost for removing carbon dioxide from the gas stream.

```
co2_cst= co2_ccst+co2_ocst/denom
if ((co2.le.co2_lim).and.(h2s.gt.h2s_lim)) then
co2_cst= amin1(co2_ccst+co2_ocst/denom,diccost+diocost/denom)

if(co2_cst.eq.diccost+diocost/denom) then
co2_ocst = diocost/denom
co2_ccst = diccost
end if

end if
```

Note: Total gas processing cost ($/MCF) for H2O is calculated by adding the operating cost and capital cost for removing water from the gas stream.

```
h2o_cst = h2o_ocst+h2o_ccst

if ((stccost+stocost).gt.(crccost+crocost)) then
ngl_ocst = crocost
ngl_ccst = crccost
type = 'cryo'
else
ngl_ocst = stocost
ngl_ccst = stccost
type = 'strt'
end if
```

Note: Total gas processing cost ($/MCF) for NGL is calculated by adding the operating cost and capital cost for removing NGL from the gas stream.

```
ngl_cst = ngl_ocst/denom+ngl_ccst
```

Step 11: **The maximum by-product revenue costs are hardwired.**

Note: If the byproduct revenue costs are greater than 1.3*(total ngl cost) then the byproduct revenue costs are set to be equal to 1.3*(total ngl cost).

```
      if (byrevs(5).gt.1.3*ngl_cst) byrevs(5) = 1.3*ngl_cst
```

Note: 'totnet' is the total gas processing cost. It comprises the difference of the sum of the gas processing cost for the removal of N2, CO2, H2O and NGL from the gas stream and the by product revenue.

```
      totnet = n2_cst+co2_cst+h2o_cst+ngl_cst-byrevs(5)
```

Step 12: The output file 'process3.out' is opened and written for checking NGL calculations.

```
      open(unit=16,file='process3.out')
      write(16,*)'ngllev =  ',ngl,
      'ngl_cst  = ',ngl_cst,'  type =  ',type

30    continue
```

Step 13: All the output files are closed.

```
      close(40)
      close(41)
      close(42)
      close(43)
      close(44)
900   format(a2,t12,f7.3,t20,f7.3,t28,f7.3,t36,f7.3,t44,f7.3,t52,f4.2)
901   format(a2,t12,f5.3,t18,f5.3,t24,f5.3,t30,f5.3,t36,f5.3)
902   format(f4.2,1x,f4.1,1x,f4.2,1x,f4.2,1x,f4.2,1x,f9.7,1x,f5.3)
903   format(1x,'gsamid: ',a11,1x,f10.3)
905   format(a8,t8, a11,t22,a11,t35,a13,t55,a3)
910   format(a7,t10,f10.4,t20,f10.4,t35,f10.4,t50,f10.4)
      return
      end
```

Note: Various subroutines called by the program (process.for) are listed below and explained in detail.

SUBROUTINE BY_REV
(This calculates the by-product revenues)

Note: It is assumed that no revenue is generated by selling CO_2, H_2S or H_2O obtained from the gas stream.

```
co2_rev = 0.000
h2s_rev = 0.000
h2o_rev = 0.00
```

Note: The ngl value is capped at 85 bbl/MMCF. The revenue associated with selling NGL is calculated below in $/MCF. 0.30 $/Gallon is the price for NGL, 42 is the conversion factor to convert from gallons to bbls and 1000 is the conversion factor to convert from MMCF to MCF.

```
if (ngl.gt.85.0) then
ngl_rev = 0.3000*85.0*42.0/1000.00
else
ngl_rev = 0.3000*ngl*42.0/1000.00
end if
```

Note: There is no revenue from N2 sales.

```
n2_rev  = 0.00
```

Note: The total by-product revenue is calculated in $/MCF.

```
byp_rev = n2_rev+co2_rev+h2s_rev+h2o_rev+ngl_rev
```

Note: The by-product revenues are updated.

```
byrevs(1) = n2_rev
byrevs(2) = co2_rev+h2s_rev
byrevs(3) = h2o_rev
byrevs(4) = ngl_rev
byrevs(5) = byp_rev

return
end
```

SUBROUTINE CALC_UCC

(This calculates the unitized capital costs in $/MCF)

Note: The input to the subroutine is total capital cost in millions of dollars, the output cost is the unitized capital cost in $/MCF. 'ac' is the annual capital charge ($10**6/year) and 'avt' is the average volume thoughput ($10**6 cubic feet/day).

```
ac  = cost*dis_rate/(1.00-exp(-dis_rate*plt_lfe))
avt = capacity*365.00*op_factor
cost = ac*(million/thousand)/(avt)
return
end
```

SUBROUTINE CAP_COST

(This calculates capital cost in 10**6 dollars and then converts it to $/MCF via the subroutine calc_ucc which assumes that the capacity is in MMCF/day.)

Note: The costs for each impurity will be generated only if impurity percentages are above the associated limits.

Note: The costs are initialized.

```
      n2_cst  = 0.00
      co2_cst = 0.00
      h2o_cst = 0.00
      ngl_cst = 0.00
      dicost  = 0.0
      stcost  = 0.0
      crcost  = 0.0
```

Note: The capital costs for N2 removal are calculated.

```
      if (n2.gt.n2_lim) then
      call nitrocc(n2_cst,captexas)
      call calc_ucc(n2_cst,captexas,dis_rate,plt_lfe,op_factor)
1000  format(' n2_cst in $/MCF',f10.5)
      end if
```

Note: The Carbon Dioxide and Hydrogen Sulfide Processing Costs are calculated. The cost for both these chemicals is stored in 'co2_cst'. The subroutine `deatrtcc' calculates the capital costs for the DEA Treating Plant. The subroutine `clauscc' calculates the capital costs for the Claus Sulfur Recovery Plant. The subroutine `directcc' calculates the capital costs for the Direct Conversion of H_2S to Sulfur (using the Chelated Iron Process).

```
      if ((co2.gt.co2_lim).and.(h2s.le.h2s_lim)) then
      call deatrtcc(acid,captexas,co2_cst)
      call calc_ucc(co2_cst,captexas,dis_rate,plt_lfe,op_factor)
      if (h2s.gt.0.000) then
      endif
      else if ((co2.gt.co2_lim).and.(h2s.gt.h2s_lim)) then
      call deatrtcc(acid,captexas,co2_cst)
      call calc_ucc(co2_cst,captexas,dis_rate,plt_lfe,op_factor)
      if (h2s.gt.0.000) then
      endif
      call clauscc(capsulf,clcost)
```

Note: Here the 'capsulf' (which is the capacity (long tons/day) for Claus Sulfur Recovery and the Direct Conversion of H2S to Sulfur (Chelated Iron Process)) is converted from long tons to MMCF. 1 long ton of sulfur equals 0.0246 MMCF. Also since the total capacity for the inlet gas is needed and not just h2s we divide by the h2s %.

```
      capacity = (capsulf/h2s)*0.024600
      call calc_ucc(clcost,capacity,dis_rate,plt_lfe,op_factor)
      if (h2s.gt.0.0) then
      endif
1002  format(' claus cap cost in $/MCF',f10.5)
      co2_cst = co2_cst + clcost
      else if ((co2.le.co2_lim).and.(h2s.gt.h2s_lim)) then
      call deatrtcc(acid,captexas,co2_cst)
      call calc_ucc(co2_cst,captexas,dis_rate,plt_lfe,op_factor)
      call clauscc(capsulf,clcost)
      capacity = (capsulf/h2s)*0.0246000
      call calc_ucc(clcost,capacity,dis_rate,plt_lfe,op_factor)
      co2_cst = co2_cst + clcost

      call directcc(capsulf,dicost)
      capacity = (capsulf/h2s)*0.0246000
      call calc_ucc(dicost,capacity,dis_rate,plt_lfe,op_factor)

1003  format(' direct cost in $/MCF',f10.5)

      else
```

Note: If both co2 and h2s are under the prescribed limits then the cost is zero.

```
      co2_cst = 0.00
      end if
```

Note: Here the capital costs of Glycol Dehydration are calculated.

```
      if (h2o.gt.h2o_lim) then
      call glycolcc(h2o_cst,captexas)
      call calc_ucc(h2o_cst,captexas,dis_rate,plt_lfe,op_factor)
1004  format(' glycol cost in $/MCF',f10.5)

      end if
```

Note: Here the total cost for NGL removal is calculated. The subroutine 'strtcc' calculates the capital costs for the Straight Refrigeration Plant. The subroutine 'cryocc' calculates the capital costs for the Cryogenic Expander Plant.

```
      if (ngl.gt.ngl_lim) then
      call strtcc(capstrt,stcost,ngl,thruput)
      call calc_ucc(stcost,capstrt,dis_rate,plt_lfe,op_factor)
      call cryocc(capcryo,crcost)
      call calc_ucc(crcost,capcryo,dis_rate,plt_lfe,op_factor)
1005  format(' straight cost in $/MCF', f10.5)
1006  format(' cryogenic cost in $/MCF', f10.5)
      end if

      return
      end
```

<u>SUBROUTINE CHECK1</u>

(This subroutine performs routine consistency checks and prints pertinent messages on the screen.)

Note: Checks are performed on capacities and thruput variables.

```
if (capst.lt.0.00) then
write(*,*)'Inlet capacity for region ',regnum,' is incorrect'
write(*,*)'Program halted'
stop
end if

if (capcr.lt.0.00) then
write(*,*)'Inlet capacity for region ',regnum,' is incorrect'
write(*,*)'Program halted'
stop
end if

if (capte.lt.0.00) then
write(*,*)'Inlet capacity for region ',regnum,' is incorrect'
write(*,*)'Program halted'
stop
end if

if (capsu.lt.0.00) then
write(*,*)'Inlet capacity for region ',regnum,' is incorrect'
write(*,*)'Program halted'
stop
end if

if (thru.lt.0.00) then
write(*,*)'Gas thruput for region ',regnum,' is incorrect'
write(*,*)'Program halted'
stop
end if

if ((op_fac.lt.0.00).or.(op_fac.gt.1.00)) then
write(*,*)'Operating factor for region ',regnum,' is incorrect'
write(*,*)'Program halted'
stop
end if

return
end
```

SUBROUTINE CHECK2

(This subroutine performs routine consistency checks and prints pertinent messages on the screen.)

Note: Checks are performed on N_2, H_2S, CO_2, H_2O and NO_2 concentrations in the gas stream for every GSAMID.

```
if ((n2.gt.1.00).or.(n2.lt.0.00)) then
write(*,*)'Incorrect % for N_2 value in reservoir # ',resnum
write(*,*)'Program halted'
stop
end if

if ((co2.gt.1.00).or.(co2.lt.0.00)) then
write(*,*)'Incorrect % for CO_2 value in reservoir # ',resnum
write(*,*)'Program halted'
stop
end if


if ((h2o.gt.1.00).or.(h2o.lt.0.00)) then
write(*,*)'Incorrect % for H_2O value in reservoir # ',resnum
write(*,*)'Program halted'
stop
end if


if ((h2s.gt.1.00).or.(h2s.lt.0.00)) then
write(*,*)'Incorrect % for H_2S value in reservoir # ',resnum
write(*,*)'Program halted'
stop
end if


if (ngl.lt.0.00) then
write(*,*)'Incorrect % for NGL value in reservoir # ',resnum
write(*,*)'Program halted'
stop
end if

return
end
```

SUBROUTINE CHECK3

(This subroutine performs routine consistency checks and prints pertinent messages on the screen.)

Note: The subroutine checks the validity of the data specified in ‘pro_reg.spc’ and ‘pro_dat.spc’ files.

```
if ((dis_rate.gt.1.00).or.(dis_rate.le.0.00)) then
write(*,*)'Discount rate =', dis_rate, 'incorrect data'
write(*,*)'Program halted'
stop
end if


if ((plt_lfe.lt.0.00)) then
write(*,*)'Plant life =', plt_lfe, 'incorrect data'
write(*,*)'Program halted'
stop
end if


if ((n2_lim.gt.1.00).or.(n2_lim.le.0.00)) then
write(*,*)'N2 limit =', n2_lim, 'incorrect data'
write(*,*)'Program halted'
stop
end if


if ((co2_lim.gt.1.00).or.(co2_lim.le.0.00)) then
write(*,*)'CO2 limit =', co2_lim, 'incorrect data'
write(*,*)'Program halted'
stop
end if

if ((h2o_lim.gt.1.00).or.(h2o_lim.le.0.00)) then
write(*,*)'H2O limit =', h2o_lim, 'incorrect data'
write(*,*)'Program halted'
stop
end if

if ((h2s_lim.gt.1.00).or.(h2s_lim.le.0.00)) then
write(*,*)'H2S limit =', h2s_lim, 'incorrect data'
write(*,*)'Program halted'
stop
end if

if (ngl_lim.lt.0.00) then
write(*,*)'NGL limit =', ngl_lim, 'incorrect data'
write(*,*)'Program halted'
stop
end if


return
end
```

SUBROUTINE CLAUSCC

(This subroutine calculates the capital costs for the Claus Sulfur Recovery Plant)

```
if (caplong.le.30.00) then
clcost = exp(0.572400*alog(caplong)-0.836800)
else
clcost = exp(0.754200*alog(caplong)-1.3122000)
end if


return
end
```

SUBROUTINE CLAUSOC

(This subroutine calculates the operating costs for the Claus Sulfur Recovery Plant)

```
clcost = exp(6.021000*exp(-0.165600*alog(caplong)))
```

Note: Here the costs in $/Long Ton are converted to $/MCF.

```
clcost = clcost/2240*2.205*34.076/22.4*28.3169*h2s/(1-h2s)
return
end
```

SUBROUTINE CRYOCC

(This subroutine calculates the capital costs for the Cryogenic Expander Plant.)

Note: The regional value for capacity (MMCFD) for the NGL Cryogenic Expander Plant is read (capinlet) from the 'pro_reg.spc' file. A different set of equations is used for a capacity less than 100 MMCFD and more than 100 MMCFD.

```
if (capinlet.lt.100.00) then
crcost = exp(0.484200*alog(capinlet)-0.096200)
else
crcost = exp(0.989300*alog(capinlet)-2.376900)
end if

return
end
```

SUBROUTINE CRYOOC

(This subroutine calculates the operating costs for the Cryogenic Expander Plant.)

Note: The following calculations (performed for a capacity of 30 MMCFD) are used to minimize costs.

```
if (capinlet.lt.30.00) then
crcost = exp(2.738000*exp(-0.253400*alog(capinlet)))
else
crcost = exp(4.341700*exp(-0.246300*alog(capinlet)))
end if
```

Note: The costs are in cents/MCF, and are converted to $/MCF.

```
crcost = crcost/100.00
return
end
```

SUBROUTINE DEATRTCC

(This subroutine calculates the capital costs for the DEA Treating Plant)

Note: Costs are calculated in dlog-dlog space and then interpolated before exponentiating, this is because in log-log space the relationships are linear hence the weighting scheme makes sense.

```
cost1   = (0.643000*alog(capinlet)-2.690000)
cost2p5 = (0.600900*alog(capinlet)-1.843700)
cost5   = (0.635000*alog(capinlet)-1.645800)
cost10  = (0.632500*alog(capinlet)-1.169000)
cost20  = (0.624400*alog(capinlet)-0.739600)
```

Note: Interpolation is performed between these values to recover the appropriate costs. The calculation is performed at total acid gas concentrations (which is the sum of the H2S and CO2 gas concentrations.)

```
if (acid.le.0.0100)  then
weightb = acid/0.0100
call wt_cst(weightb,0.00,cost1,co2_cst)

else if (acid.le.0.02500) then
weightb = (acid-0.0100)/(0.02500-0.0100)
call wt_cst(weightb,cost1,cost2p5,co2_cst)

else if (acid.le.0.0500)  then
weightb = (acid-0.02500)/(0.0500-0.02500)
call wt_cst(weightb,cost2p5,cost5,co2_cst)

else if (acid.le.0.1000)  then
weightb = (acid-0.0500)/(0.1000-0.0500)
call wt_cst(weightb,cost5,cost10,co2_cst)

else if (acid.le.0.2000)  then
weightb = (acid-0.1000)/(0.2000-0.1000)
call wt_cst(weightb,cost10,cost20,co2_cst)

else
```

Note: If acid gas concentration is greater than 0.20 (i.e., 20%), the cost (in log form) for the 20% level is used.

```
co2_cst = cost20
end if
```

Note: The costs are exponentiated to get them in `normal' (non log-log) space.

```
co2_cst = exp(co2_cst)

return
end
```

SUBROUTINE DEATRTOC

(This subroutine calculates the operating costs for the DEA Treating Plant)

Note: Costs are calculated in dlog-dlog space and then interpolated before exponentiating this is because in log-log space the relationships are linear and the weighting scheme makes sense in this space.

```
cost1   = (-0.354400*alog(thru)-2.483400)
cost5   = (-0.312600*alog(thru)-2.397100)
cost10  = (-0.274500*alog(thru)-2.242500)
cost20  = (-0.176000*alog(thru)-2.114100)
```

Note: Interpolation is performed between these values to recover the appropriate costs. The calculation is performed at total acid gas concentrations (which is the sum of the H2S and CO2 gas concentrations.)

```
if (acid.le.0.0100)  then
weightb = acid/0.0100
call wt_cst(weightb,0.00,cost1,co2_cst)

else if (acid.le.0.0500)  then
weightb = (acid-0.0100)/(0.0500-0.0100)
call wt_cst(weightb,cost1,cost5,co2_cst)

else if (acid.le.0.1000)  then
weightb = (acid-0.0500)/(0.1000-0.0500)
call wt_cst(weightb,cost5,cost10,co2_cst)

else if (acid.le.0.2000)  then
weightb = (acid-0.1000)/(0.2000-0.1000)
call wt_cst(weightb,cost10,cost20,co2_cst)

else
```

Note: If acid gas concentration is greater than 0.20 (i.e., 20%), the cost (in log form) for the 20% level is used.

```
co2_cst = cost20
end if
```

Note: The costs are exponentiated to get them in `normal' (non log-log) space.

```
co2_cst = exp(co2_cst)

return
end
```

SUBROUTINE DIRECTCC

(This subroutine calculates the capital costs for the Direct Conversion process of H_2S to Sulfur (Chelated Iron Process))

```
      dicost = exp(1.170700*alog(caplong)-1.199900)

      return
      end
```

SUBROUTINE DIRECTOC

(This subroutine calculates the operating costs for the Direct Conversion process of H_2S to Sulfur (Chelated Iron Process))

```
      dicost = exp(-0.315600*alog(caplong)+6.302400)
```

Note: Here the costs in $/Long Ton are converted to $/MCF.

```
      dicost = dicost/2240*2.205*34.076/22.4*28.3169*h2s/(1-h2s)
      return
      end
```

SUBROUTINE OP_COST

(This subroutine calculates operating costs)

Note: Operating costs are calculated in $/MCF.

Note: If the impurity is above the acceptable limit, costs are calculated using the suggested technology given in reference (2).

```
n2_cst  = 0.00
co2_cst = 0.00
h2o_cst = 0.00
ngl_cst = 0.00
stcost  = 0.0
crcost  = 0.0
```

Note: Operating costs are calculated for the Nitrogen Rejection Plant.

Note: The costs are in cents/MCF and are converted to $/MCF.

```
if (n2.gt.n2_lim) then
call nitrooc(n2_cst,captexas)
n2_cst = n2_cst/100.00
```

Note: Fuel costs are taken into account.

```
n2_cst = n2_cst*2.0
end if
```

Note: The Carbon Dioxide and Hydrogen Sulfide Processing Costs are calculated. The cost for both these chemicals is stored in 'co2_cst'. The subroutine `deatrtoc' calculates the operating costs for the DEA Treating Plant. The subroutine `clausoc' calculates the operating costs for the Claus Sulfur Recovery Plant. The subroutine `directoc' calculates the operating costs for the Direct Conversion of H_2S to Sulfur (using the Chelated Iron Process).

```
if ((co2.gt.co2_lim).and.(h2s.le.h2s_lim)) then
call deatrtoc(acid,thruput,co2_cst)
```

Note: Fuel costs are taken into account.

```
co2_cst = co2_cst*2.0
```

```
else if ((co2.gt.co2_lim).and.(h2s.gt.h2s_lim)) then
call deatrtoc(acid,thruput,co2_cst)
call clausoc(capsulf,clcost,h2s)
co2_cst = co2_cst + clcost
```

Note: Fuel costs are taken into account.

```
co2_cst = co2_cst*2.0

else if ((co2.le.co2_lim).and.(h2s.gt.h2s_lim)) then
call deatrtoc(acid,thruput,co2_cst)
call clausoc(capsulf,clcost,h2s)
co2_cst = co2_cst + clcost
call directoc(capsulf,dicost,h2s)
```

Note: Fuel costs are taken into account.

```
co2_cst = co2_cst*2.0
dicost  = dicost*2.0
else
```

Note: If both co2 and h2s are under the prescribed limits then the cost is zero.

```
co2_cst = 0.00
end if
```

Note: Here the operating costs of Glycol Dehydration are calculated.

```
if (h2o.gt.h2o_lim) then
call glycoloc(h2o_cst,captexas)
```

Note: Fuel costs are taken into account.

```
h2o_cst = h2o_cst*2.0
end if
```

Note: Here the total cost for NGL removal is calculated. The subroutine `strtoc' calculates the operating costs for the Straight Refrigeration Plant. The subroutine `cryooc' calculates the operating costs for the Cryogenic Expander Plant.

```
if (ngl.gt.ngl_lim) then
call strtoc(capstrt,stcost,ngl,thruput)
call cryooc(capcryo,crcost)
```

Note: Fuel costs are taken into account.

```
stcost = stcost * 2.0
crcost = crcost * 2.0
end if
return
end
```

SUBROUTINE STRTCC

(This subroutine calculates the capital costs for the Straight Refrigeration Plant.)

Note: Costs are first used in log-log space as opposed to normal space and then converted back later. This is done because cost equations for 12,24,48,96,192 gpm (gallons per minute) were developed by using the difference in the natural logs of the capital costs from 3 to 6 gpm and applying this difference to create the 12,24,48, 96,192 gpm cases. The difference was adjusted between each case by the same ratio that the difference changed from between the 1.5 minus 3.0 vs. the 6.0 minus 3.0 case.

```
stcost192 = (1.4396*alog(capinlet)-1.0789)

stcost96 = (1.194*alog(capinlet)-0.8611)

stcost48 = (0.9868*alog(capinlet)-0.7031)

stcost24 = (0.8139*alog(capinlet)-0.6011)

stcost12 = (0.6724*alog(capinlet)-0.5536)

stcost6 = (0.561200*alog(capinlet)-0.562800)

stcost3 = (0.481300*alog(capinlet)-0.636600)

stcost1 = (0.437900*alog(capinlet)-0.792900)
```

Note: The gallons per minute (GPM) is calculated and interpolated between the three costs given above. For NGL (natural gas liquids), 'thruput' is gas thruput (MMCFD) and 'ngl' is ngl in bbl/MMCF.

Note: NGL is capped at 85 bbl/MMCF.

```
if (ngl.gt.85.0) then
gpm = thruput*85.0*42.0/24.00/60.00
else
gpm = thruput*ngl*42.0/24.00/60.00
end if
```

Note: The costs are interpolated based on the gpm value.

```
if (gpm.le.1.500) then
weightb = gpm/1.500
call wt_cst(weightb,0.00,stcost1,stcost)
```

```
else if (gpm.le.3.00) then
weightb =  (gpm-1.500)/(3.00-1.500)
call wt_cst(weightb,stcost1,stcost3,stcost)
else if (gpm.le.6.00) then
weightb = (gpm-3.00)/(6.00-3.00)
call wt_cst(weightb,stcost3,stcost6,stcost)
else if (gpm.le.12.00) then
weightb = (gpm-6.00)/(12.00-6.00)
call wt_cst(weightb,stcost6,stcost12,stcost)
else if (gpm.le.24.00) then
weightb = (gpm-12.00)/(24.00-12.00)
call wt_cst(weightb,stcost12,stcost24,stcost)
else if (gpm.le.48.00) then
weightb = (gpm-24.00)/(48.00-24.00)
call wt_cst(weightb,stcost24,stcost48,stcost)
else if (gpm.le.96.00) then
weightb = (gpm-48.00)/(96.00-48.00)
call wt_cst(weightb,stcost48,stcost96,stcost)
else if (gpm.le.192.00) then
weightb = (gpm-96.00)/(192.00-96.00)
call wt_cst(weightb,stcost96,stcost192,stcost)
else
stcost = stcost192
end if
```

Note: The costs are exponentiated to bring them to normal (non log) space.

```
stcost = exp(stcost)


return
end
```

SUBROUTINE STRTOC

(This subroutine calculates the operating costs for the Straight Refrigeration Plant.)

```
if (capinlet.le.6.00) then
stcost192 = 0.3667*capinlet**(-.5809)
else
stcost192 = 0.2317*capinlet**(-0.3116)
end if

if (capinlet.le.6.00) then
stcost96 = 0.3406*capinlet**(-0.5761)
else
stcost96 = 0.2177*capinlet**(-0.3134)

end if

if (capinlet.le.6.00) then
stcost48 = 0.3144*capinlet**(-0.5706)
else
stcost48 = 0.2038*capinlet**(-0.3155)
end if

if (capinlet.le.6.00) then
stcost24 = 0.2883*capinlet**(-0.5642)
else
stcost24 = 0.1898*capinlet**(-0.3179)
end if

if (capinlet.le.6.00) then
stcost12 = 0.2622*capinlet**(-0.5566)
else
stcost12 = 0.1759*capinlet**(-0.3207)
end if

if (capinlet.le.6.00) then
stcost6 = 0.2361*capinlet**(-0.5475)
else
stcost6 = 0.162*capinlet**(-0.3241)
end if

if (capinlet.le.6.00) then
stcost3 = 0.21*capinlet**(-0.5363)
else
stcost3 = 0.1481*capinlet**(-0.3282)
end if

if (capinlet.le.6.00) then
stcost1 = 0.2035*capinlet**(-0.5782)
else
stcost1 = 0.1293*capinlet**(-0.3159)
end if
```

Note: The gallons per minute (GPM) is calculated and interpolated between the three costs given above. For NGL (natural gas liquids), 'thruput' is gas thruput (MMCFD) and 'ngl' is ngl in bbl/MMCF.

Note: NGL is capped at 85 bbl/MMCF.

```
if (ngl.gt.85.0) then
gpm = thruput*85.0*42.0/24.00/60.00
else
gpm = thruput*ngl*42.0/24.00/60.00
end if
```

Note: The costs are interpolated based on the gpm value.

```
if (gpm.le.1.500) then
weightb = gpm/1.500
call wt_cst(weightb,0.00,stcost1,stcost)
else if (gpm.le.3.00) then
weightb =  (gpm-1.500)/(3.00-1.500)
call wt_cst(weightb,stcost1,stcost3,stcost)
else if (gpm.le.6.00) then
weightb = (gpm-3.00)/(6.00-3.00)
call wt_cst(weightb,stcost3,stcost6,stcost)
else if (gpm.le.12.00) then
weightb = (gpm-6.00)/(12.00-6.00)
call wt_cst(weightb,stcost6,stcost12,stcost)
else if (gpm.le.24.00) then
weightb = (gpm-12.00)/(24.00-12.00)
call wt_cst(weightb,stcost12,stcost24,stcost)
else if (gpm.le.48.00) then
weightb = (gpm-24.00)/(48.00-24.00)
call wt_cst(weightb,stcost24,stcost48,stcost)
else if (gpm.le.96.00) then
weightb = (gpm-48.00)/(96.00-48.00)
call wt_cst(weightb,stcost48,stcost96,stcost)
else if (gpm.le.192.00) then
weightb = (gpm-96.00)/(192.00-96.00)
call wt_cst(weightb,stcost96,stcost192,stcost)
else
stcost = stcost192
end if

return
end
```

SUBROUTINE WT_CST

(This subroutine is used in interpolation schemes)

```
if ((weightb.gt.1.00).or.(weightb.lt.0.00)) then
stop
end if
weighta = 1.00-weightb
cost    = weighta*costa+weightb*costb

return
end
```

SUBROUTINE GLYCOLCC
(This subroutine calculates the capital costs for glycol dehydration.)

```
h2o_cst = exp(0.625700*log(captexas)-3.635300)
return
end
```

SUBROUTINE GLYCOLOC

(This subroutine calculates the operating costs for glycol dehydration.)

```
      h2o_cst = exp(-0.387000*alog(captexas)-2.727600)*0.93300

      return
      end
```

SUBROUTINE NITROCC

(This subroutine calculates the capital costs for the Nitrogen Rejection Plant.)

```
      n2_cst=exp(0.457500*log(captexas)+0.359700)
      return
      end
```

SUBROUTINE NITROOC

(This subroutine calculates the operating costs for the Nitrogen Rejection Plant.)

```
n2_cst=exp(3.433900*exp(-0.129100*log(captexas)))

return
end
```

Table of Contents

MAKEBIN

CALLS:	ERRMSG (Prints out errors and warnings)
READS:	‘spec.dat’ (Specifies the discovered resource to be included in the bank files) ‘*.prd’ (Contains production and operating costs) ‘*.dec’ (Contains reservoir decisions) ‘*.gsm’ (Contains the discovered US database)
CREATES:	‘undb.bnk’ (This is the undiscovered reservoir data bank) ’disb.bnk’ (This is the undiscovered reservoir data bank) ‘disb.tcp’ (This is the discovered reservoir data bank) ‘undb.tcp’ (This the discovered reservoir data bank)
MAIN THEME:	This program creates the data bank (UNDB.BNK, UNDB.TCP, for undiscovered reservoir and DISB.BNK, DISB.TCP for discovered reservoirs).

Step 1: **The screening price (scrprc) is set to 15.**

Note: The screening price is used to screen reservoirs that pass the test described in Step 6.

```
      scrprc=15.00
```

Step 2: **'Spec.dat' is opened in the 'news' directory.**

Note: This file is used in the creation of the undiscovered/discovered "data bank" files. It contains information about file names (such as ‘ofn1’, ‘ofn2’ which may be specified as ‘undb.bnk’ and ‘undb.tcp’ etc.) that are used to create the data bank.

```
      Open(11,file='news\spec.dat')
      read(11,101) ofn1,ofn2
101   format(a15,t24,a15)
```

Step 3: **The files ‘ofn1’ (i.e., ‘undb.bnk’/’disb.bnk’) and ‘ofn2’ (i.e., ‘disb.tcp’/‘undb.tcp’) are opened in binary format.**

Note: The filenames ‘*.prd’ (ifn1), ‘*.dec’ (ifn3), ‘*.gsm’ (ifn5) are read and are used to read and create the databank files.

```
      open(34,file=ofn1,form='BINARY')
      open(35,file=ofn2,form='BINARY')

500   read(11,110,end=9000) ifn1,ifn3,ifn5
110   format(a15,t20,a15,t39,a15)
      write(*,102) ifn1
102   format(' ',a15)
      open(24,file=ifn1)
      open(25,file=ifn3)
      if(ifn5.ne.'              ') then
      ifc5=1
      open(28,file=ifn5)
      else
      ifc5=0
      endif
```

Step 4: **The following section of code is read only when the undiscovered databank is created.**

Note: The read statement reads the GSAMID and the number of undiscovered accumulations from the ‘*.GSM’ (undiscovered) files.

```
1000  if(ifc5.ne.0) then
      read(28,1001,end=1010) flgx,nrrr
1001  format(a11,t92,i9)
      go to 1020
1010  flgx='-x-x-x-x-x-'
1020  continue
      endif
```

Note: This loop is for two technologies (current and advanced).

```
      do 1500 ix=1,2
      if(ix.eq.1) then
      iy=iy1
      else
      iy=iy2
      endif
```

Step 5: **Here the ‘*.DEC’ and ‘*.PRD’ file are read.**

Note: The variable ‘ctch(1) is for reading the header of the *.DEC file. The ‘DO loop’ 1200 reads all the entries for the ‘*.PRD’ file (18 entries). Operating costs ($/MCF) and production rates (BCF/yr) are read for every reservoir for every pay grade, development type and technology type.

```
      if1=24
      if2=25

      read(if1,1129,end=2100) ctch(1)
1129  format(a1)
      read(if1,1129,end=2100) ctch(1)
      do 1200 ii=1,9
      read(if2,7132,end=2000) flg,ipn,c1,rsty,ctst,f1,
      ctch(ii),d(ii),copt(ii),imyr(ii),
      (oam(i,ii),i=1,40)
      read(if2,7132,end=2000) flg,ipn,c1,rsty,ctst,f1,
      ctch(ii),d(ii),copt(ii),imyr(ii),
      (prd(i,ii),i=1,40)
```

Note: For undiscovered reservoirs the '*.GSM', '*.PRD', and '*.DEC' files are checked for consistency in terms of the location of GSAMIDs. If there is an inconsistency then a fatal error message is printed.

```
      if(ifc5.eq.0) flgx=flg
      if(flgx.ne.flg) then
      write(*,9131) flg,flgx
9131  format(' tcp: ',2(1x,a20))
      call errmsg(4,201)
      endif

1132  format(a11,t1,a2,a1,i1,a4,i3,12x,a1,i2,1x,a1,6x,i2,40f8.3)
7132  format(a11,t1,a2,a1,i1,a4,i3,12x,a1,i2,1x,a1,6x,i2,1x,
40    (f9.4,1x))
```

Step 6: **This section of the code is used to store only those reservoirs in the data bank file for which the production rate times the screening price is greater than the operating and maintenance costs.**

```
      jmyr=min0(max0(imyr(ii),1),40)
      ij=jmyr
      do 1135 i=ij,1,-1
      jmyr=i
      if((prd(i,ii).gt.0.0).and.((prd(i,ii)*scrprc).ge.oam(i,ii)))
      go to 1136
1135  continue
      jmyr=1
```

Step 7: **Entries into 'undb.tcp/disb.tcp' files are finally written.**

Note: The production and operating costs entries for every GSAMID, paygrade, development type and technology are written.

```
1136    imyr(ii)=jmyr
        write(35) ipn,c1,rsty,ctst,f1,ctch(ii),d(ii),copt(ii),imyr(ii),
        (prd(i,ii),i=1,jmyr),(oam(i,ii),i=1,jmyr)
1200    continue
        ipno=ipn
        c1o=c1
        rstyo=rsty
        ctsto=ctst
        f1o=f1
1201    format(' flg: ',a11,3(1x,i3),1x,a2,a1,i1,a3,i3)
```

Step 8: **The '*.DEC' file is read.**

Note: Various variables are read:

- GSAM supply region (ipn)
- Resource status, i.e., discovered, undiscovered, or undeveloped (c1)
- resource type (rsty)
- USGS play name (ctst)
- Field size class for undiscovered accumulations and reservoir counter for discovered reservoirs (f1)
- drilling feet per well (idpth)
- water depth (ih2o)
- year window opens (iwin)
- 1 character code for technology (C, A, or M) (ctch)
- 1 character code for development option (P,I,R) (copt)
- gas reserves (xres)
- original gas in place (xogip)
- number of wells (xnw)
- MASP (xmasp)
- total capital costs (xtcap)
- present value of production (xpvp)
- present value of expenses (xpvec)
- present value of non-drilling costs (xpvndc)
- present value of drilling costs (xpvdc)
- present value of taxes (xpvtax)
- increase in present value of expenses costs when gas prices rise from $2.00 to $5.00 (xpcec)
- like xpcec but for total investment costs (xpctc)
- like xpcec but for taxes (xpctax)
- increase in NPV with $1.00 decrease in drilling cost (xpvds)
- increase in NPV iwth $1.00 decrease in non-drilling cost (xpvnds)

```
        do 1300 ii=1,9
        read(if1,1131,end=2100) ipn,c1,rsty,ctst,f1,
        ctch(ii),d(ii),copt(ii),xres,xogip,
         xnw,xmasp,xtcap,xpvp,xpvec,xpvtc,xpvdc,xpvndc,xpvtax,
         xpcec,xpctc,xpctax,xpvds,xpvnds,idpth,ih2o,iwin,ipryr

1131     format(a2,a1,i1,a4,i3,11x,a,1x,i1,1x,a,3x,f7.1,1x,f7.1,1x,
         f6.0,1x,f6.2,1x,f8.1,6(f9.3,1x),5(f8.3,1x),f6.0,1x,f6.0,
         1x,i3,8x,i3)
```

Note: Consistency checks are performed between the 'undb/disb.bnk' files and the 'undb/disb.tcp' files. If there is an inconsistency, a fatal error message is printed.

```
         if((ipn.ne.ipno).or.(c1.ne.c1o).or.(rsty.ne.rstyo).or.
         (ctst.ne.ctsto).or.(f1.ne.f1o)) then
         write(*,9132) ipn,c1,rsty,ctst,f1,flgx
9132     format(' bnk: ',a2,a1,i1,a4,i3,(1x,a20))
         call errmsg(4,201)
         endif
```

Step 9: Various variables are written to the databank 'undb/disb.bnk' files.

Note: The variables include:

- GSAM supply region (ipn)
- region status, i.e., discovered, undiscovered, or undeveloped (c1)
- resource type (rsty)
- USGS play name (ctst)
- field size class for undiscovered accumulations and reservoir counter for discovered reservoirs (f1)
- drilling feet per well (idpth)
- water depth (ih2o)
- year window opens (iwin)
- 1 character code for technology (C, A, or M) (ctch)
- 1 character code for development option (P,I,R) (copt)
- gas reserves (xres)
- original gas in place (xogip)
- number of wells (xnw)
- MASP (xmasp)
- total capital costs (xtcap)
- present value of production (xpvp)
- present value of expenses (xpvec)
- present value of non-drilling costs (xpvndc)
- present value of drilling costs (xpvdc)
- present value of taxes (xpvtax)
- increase in present value of expenses costs when gas prices rise from \$2.00 to \$5.00 (xpcec)

- like xpcec but for total investment costs (xpctc)
- like xpcec but for taxes (xpctax)
- increase in NPV with $1.00 decrease in drilling cost (xpvds)
- increase in NPV iwth $1.00 decrease in non-drilling cost (xpvnds)
- number of undiscovered accumulations (nrrr)

```
      if(ifc5.ne.0) then
```

Note: Here the ‘undb.bnk’ file is written to.

```
      write(34) ipn,c1,rsty,ctst,f1,
      ctch(ii),d(ii),copt(ii),xres,xogip,
      xnw,xmasp,xtcap,xpvp,xpvec,xpvtc,xpvdc,xpvndc,xpvtax,
      xpcec,xpctc,xpctax,xpvds,xpvnds,idpth,ih2o,
      iwin,ipryr,nrrr
      else
```

Note: Here the ‘disb.bnk’ file is written to.

```
      write(34) ipn,c1,rsty,ctst,f1,
      ctch(ii),d(ii),copt(ii),xres,xogip,
      xnw,xmasp,xtcap,xpvp,xpvec,xpvtc,xpvdc,xpvndc,xpvtax,
      xpcec,xpctc,xpctax,xpvds,xpvnds,idpth,ih2o,iwin,
      ipryr
      endif
1300  continue
1500  continue
      go to 1000
```

Note: Error messages are printed in case of inconsistencies.

```
2000  call errmsg(3,102)
      go to 2200

2100  ii=1
      read(if2,1132,end=2200) flg
      print *, 'flg', flg
      call errmsg(3,103)
      go to 2200
```

Note: The files are closed.

```
2200  close(24)
      close(25)
      if(ifc2.eq.1) then
      close(26)
      close(27)
      endif
      if(ifc5.eq.1) then
      close(28)
      endif
      go to 500
```

Step 10: The subroutine ends.

```
9000  stop
      end
```

SUBROUTINE EXDVI3

CALLED BY: EXDVSO (The output is calculated and printed in output files.)

CALLS: GETT (Estimates the time required for each phase of the subroutine.)
ERRMSG (Prints out errors and warnings)

READS: **‘**und.tcp’ (Undiscovered bank file.)
‘dis.tcp’ (Discovered bank file.)

CREATES: ‘undb.tcp’ (Undiscovered binary file.)

MAIN THEME: This routine reads the data files if they are available in ASCII files. It stores operating costs and production rates for all reservoirs for all development types, paygrades and technology options. It also converts ASCII entries into binary entries and stores the information in ‘undb.tcp’ and ‘disb.tcp’ file. It stores information in binary format for reservoirs for which revenue, calculated at some screening gas price (‘Scrprc’ in $/MCF read from the ‘gen_tml.spc’ file) for the reservoir, is higher than the operational and maintenance cost. Currently this option is not tested in the E&P module.

Step 1: The 'und.tcp' and the 'undb.tcp' files are opened.

Note: The flags showing which combinations have been read in are initialized.

```
1100    if(iocde.gt.1) call errmsg(4,489)
        open(24,file='news\und.tcp')
        open(34,file='news\undb.tcp',form='BINARY')

        v=1
        ii=0
1110    ii=ii+1
        if(ii.gt.9) then

        ii=1
        v=v+1
        if(v.gt.mxnefl) call errmsg(4,490)
        endif
```

Step 2: The 'und.tcp' file is read.

Note: Information is stored in binary format for reservoirs for which revenue, calculated at some screening gas price (‘Scrprc’ in $/MCF read from the ‘gen_tml.spc’ file) for the reservoir, is higher than the operational and maintenance cost.

```
        read(24,1131,end=1210) ipn,rsty,ctst,f1,ctch,d,copt,imyr,
        (oam(i),i=1,40)
        read(24,1131,end=1210) ipn,rsty,ctst,f1,ctch,d,copt,imyr,
        (prd(i),i=1,40)
1131    format(a2,1x,i1,a4,i3,12x,a1,i2,1x,a1,6x,i2,1x,
        40(f9.4,1x))

        imyr=min0(max0(imyr,1),40)
        ij=imyr
        do 1125 i=ij,1,-1
        imyr=i
        if((prd(i).gt.0.0).and.((prd(i)*scrprc).ge.oam(i))) go to 1126
1125    continue
        imyr=1
1126    write(34)       ipn,rsty,ctst,f1,ctch,d,copt,imyr,
        (prd(i),i=1,imyr),(oam(i),i=1,imyr)

        go to 1110
```

Step 3: The 'und.tcp' file is closed.

```
1210    close(24)

        call gett(tmes(6),tmea(6),1)
        call errmsg(1,914)
```

Step 4: The development Profiles for known fields are read in from the 'dis.tcp' file.

Note: Information is stored in binary format for reservoirs for which revenue, calculated at some screening gas price (‘Scrprc’ in $/MCF read from the ‘gen_tml.spc’ file) for the reservoir, is higher than the operational and maintenance cost.

```
        call gett(tmes(7),tmea(7),0)
1300    open(25,file='news\dis.tcp')

        ii=0
1310    ii=ii+1
        if(ii.gt.9) then
        ii=1
        v=v+1
        endif
        read(25,1131,end=1350) ipn,rsty,ctst,f1,ctch,d,copt,imyr,
        (oam(i),i=1,40)
        read(25,1131,end=1350) ipn,rsty,ctst,f1,ctch,d,copt,imyr,
        (prd(i),i=1,40)
        ipn='  '
        imyr=min0(max0(imyr,1),40)
        ij=imyr
        do 1325 i=ij,1,-1
        imyr=i
        if((prd(i).gt.0.0).and.((prd(i)*scrprc).ge.oam(i))) go to 1326
1325    continue
        imyr=1
1326    write(34)       ipn,rsty,ctst,f1,ctch,d,copt,imyr,
        (prd(i),i=1,imyr),(oam(i),i=1,imyr)

        go to 1310

1350    close(25)
        endfile 34
        close(34)
        v=v-1
        if(v.ne.nefl) call errmsg(4,605)

        call gett(tmes(7),tmea(7),1)

        return
        end
```

PROGRAMMER'S GUIDE FOR THE DEMAND AND INTEGRATING (D&I) MODULE OF THE GAS SYSTEMS ANALYSIS MODEL (GSAM)

FINAL REPORT

Volume IIId – D&I Programmer's Guide

For:

U.S. Department of Energy
National Energy Technology Laboratory
Morgantown, West Virginia
Under Contract Number: DE-AC21-92MC28138

By:

ICF Consulting, Inc.
Fairfax, Virginia

February 2001

TABLE OF CONTENTS

LIST OF FIGURES

D&I PROGRAMMER'S GUIDE GENERAL SETUP

This document provides a detailed explanation of all the major subroutines in the Demand and Integrating (D&I) module of GSAM. In the next few pages the basic structure of the D&I module is explained, followed by an explanation of the structure of this document and finally an explanation of the D&I subroutines.

The Three D&I Executables

The D&I module has three main FORTRAN executables which perform primary functions. The three main programs are INTMGN.EXE, INTRPT.EXE, and INTRVS.EXE.

INTMGN.EXE: This program translates demand, pipeline, storage and other input data into MPS format to be used as input to the LP solver. In particular, it calls other routines which:
a) read in supply, demand, and transportation input parameters,
b) calculate supply and demand curves,
c) write out constraints (e.g., material balance, demand convexity, transport capacity) in MPS format to be used as input to the LP solver

After INTMGN.EXE, TCOMB.BAT is run to concatenate the MPS files into one file (GASALL.MPS). The LP solver is then run using this file and produces an optimal solution.

INTRPT.EXE: This program creates supply and demand output reports from the LP optimal solution. The key output files produced by INTRPT.EXE are GSAMSLN.FLE (main supply and demand output file) and GSAMSLN.RPT (transportation output file).

INTRVS.EXE: This program reads in the optimal solution from the LP solver and translates the coefficients into gas supply prices written to GASPRC.NEW for each region and year. Prices derived from the dual values to the material balance constraints are written to DUAL_PRC.SPC for each region, year, and season.

ORGANIZATION OF THIS REPORT

This document is coherently structured with important routines (over and above the three main routines discussed above) separated by labeled tabs. The discussion within each tab contains the main routine (for which the tab is specified) and may also contain other subroutines which it calls. Tabs have been made for the following FORTRAN programs and appear in the document in the following sequence.

Organization of Subroutine Descriptions

Detailed descriptions of each subroutine adhere to the following format.

a) Before the explanations for the code begin there are five subheadings
   i) CALLED BY: Referring to subroutines that call the subroutine in question,
   ii) CALLS: Referring to other subroutines, that the subroutine in question calls,
   iii) READS: Referring to input files read in by the subroutine in question,
   iv) CREATES: Referring to output files created by the subroutine in question, and
   v) MAIN THEME: Providing a brief synopsis of the subroutine in question.

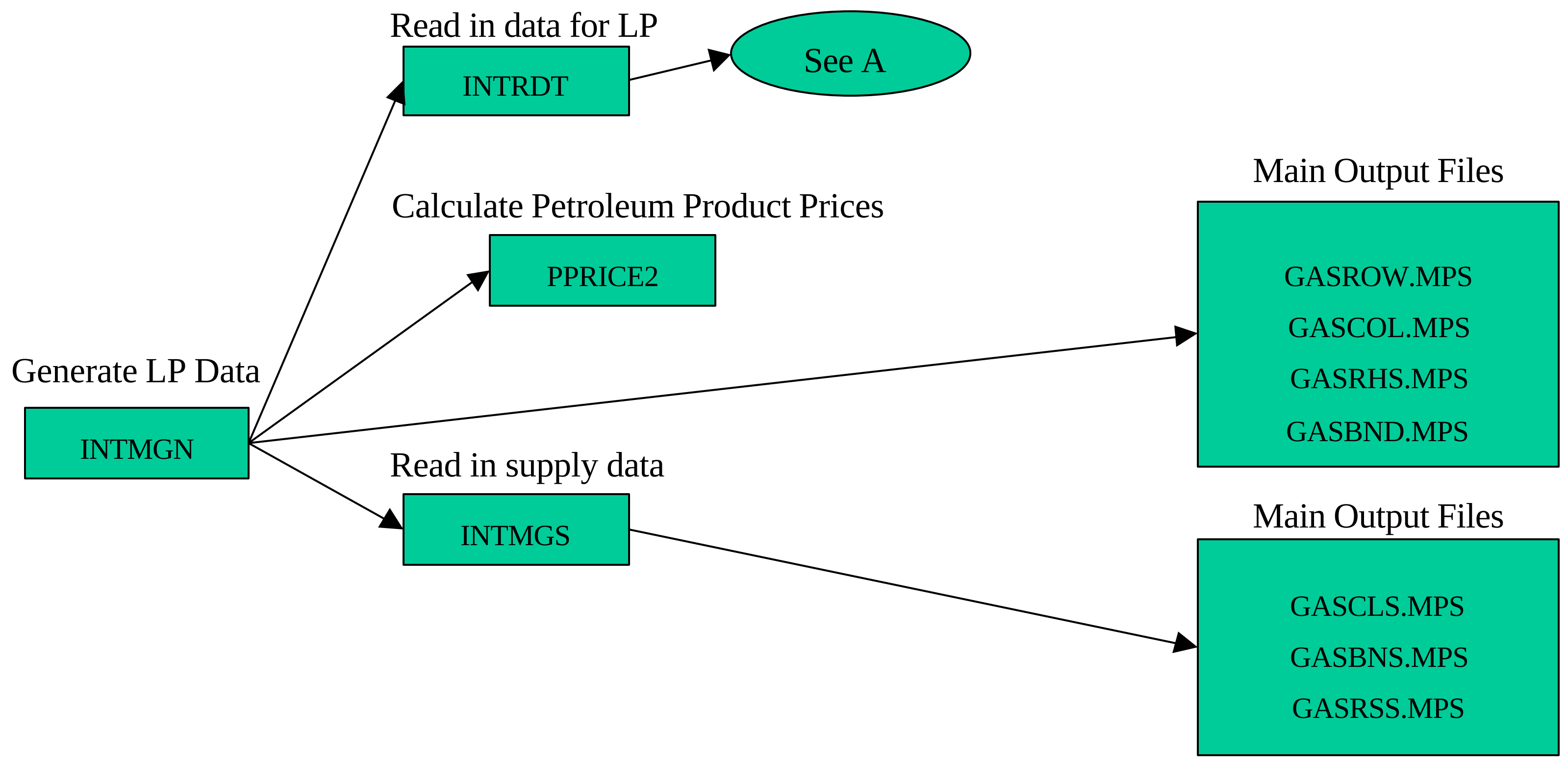
b) These five headings may not all appear in each subroutine. For example, if a subroutine does not create any output files, there will not be any subheading ‘CREATES:’.

c) These subheadings are followed by detailed explanations for the code line by line. Most of the code is explained in steps, i.e., the explanation for a group of related code is delegated to a single step. Between steps if a certain section of code needs an explanation a ‘Note’ is inserted with an appropriate description.

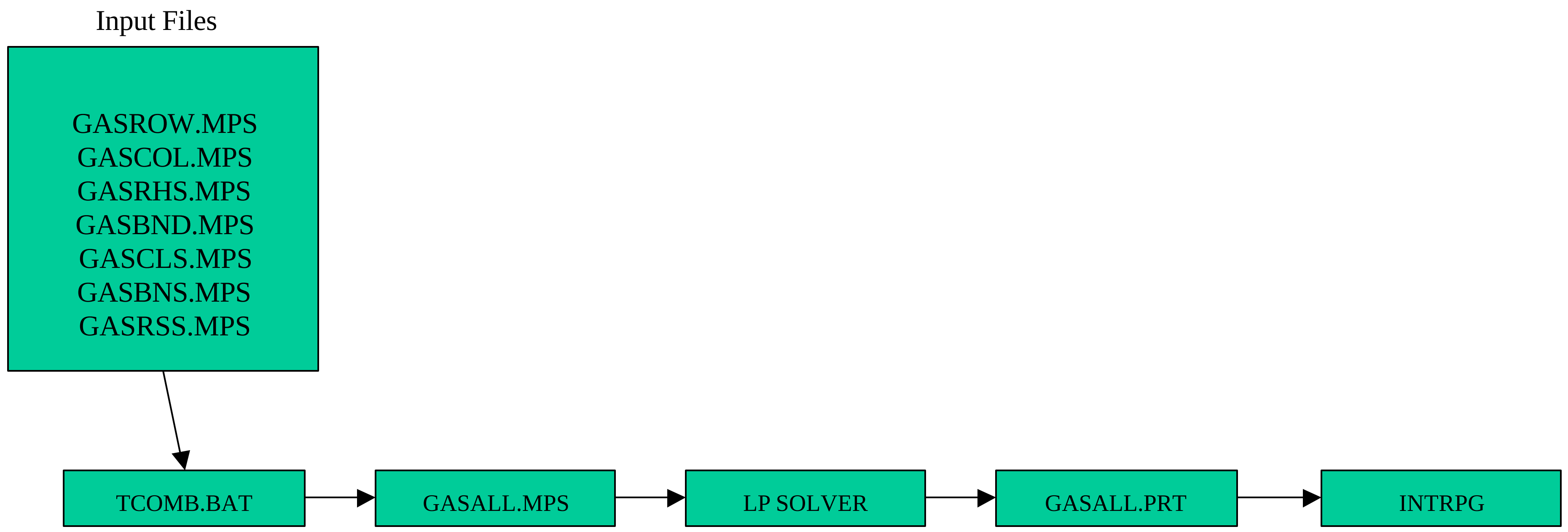
Note: The actual format of the FORTRAN programs has been modified for ease of viewing in this report. This has resulted for example, in such things as continuation markers (normally in column 6) aligning with statement labels (normally in columns 2-5). These and other related changes are mostly cosmetic but should be taken into account when reading this document.

STEP 1: GENERATE LINEAR PROGRAM DATA

EXECUTABLE: INTMGN.EXE

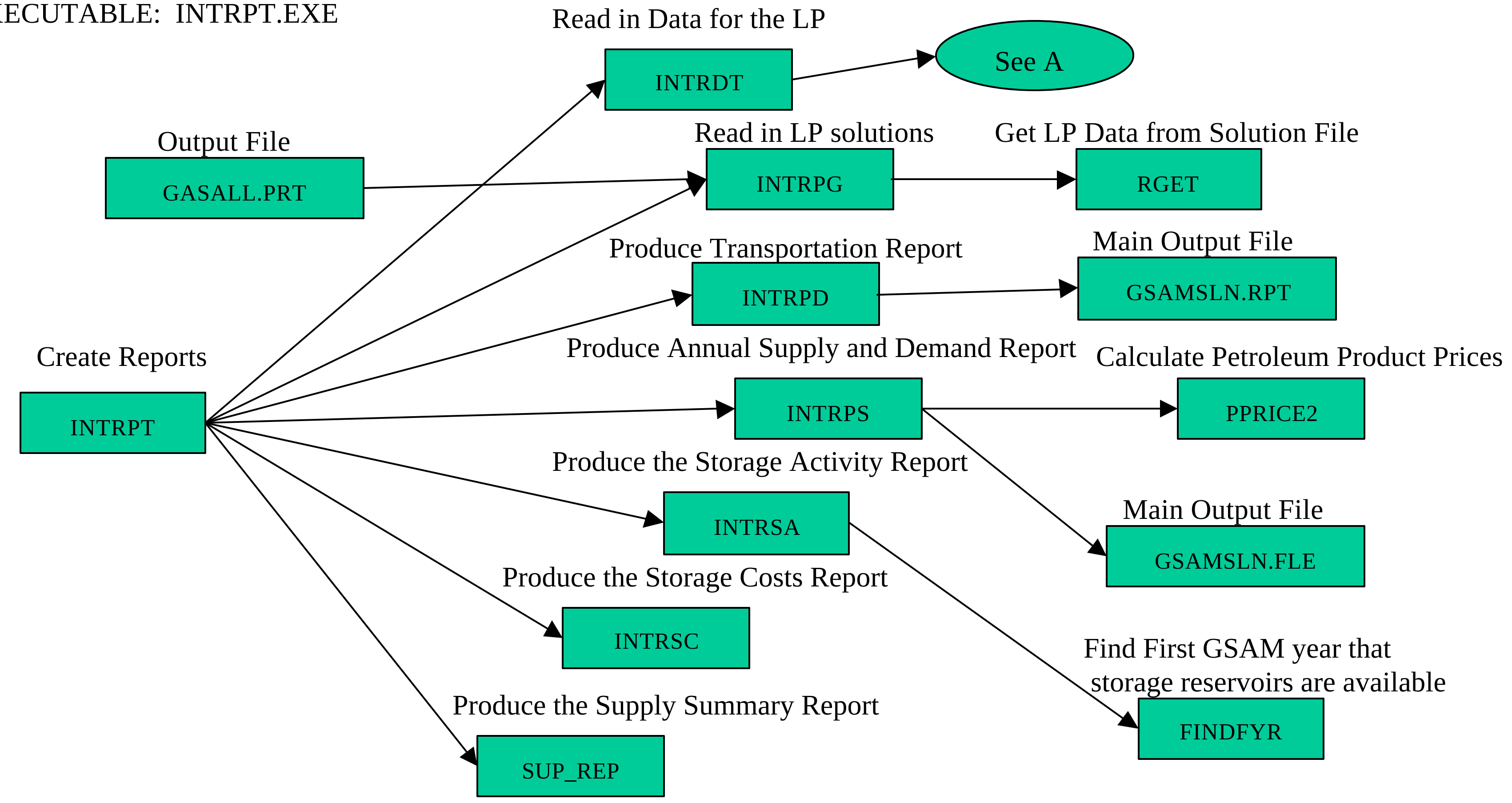


STEP 2: CONSOLIDATE LINEAR PROGRAM DATA AND RUN LINEAR PROGRAM SOLVER



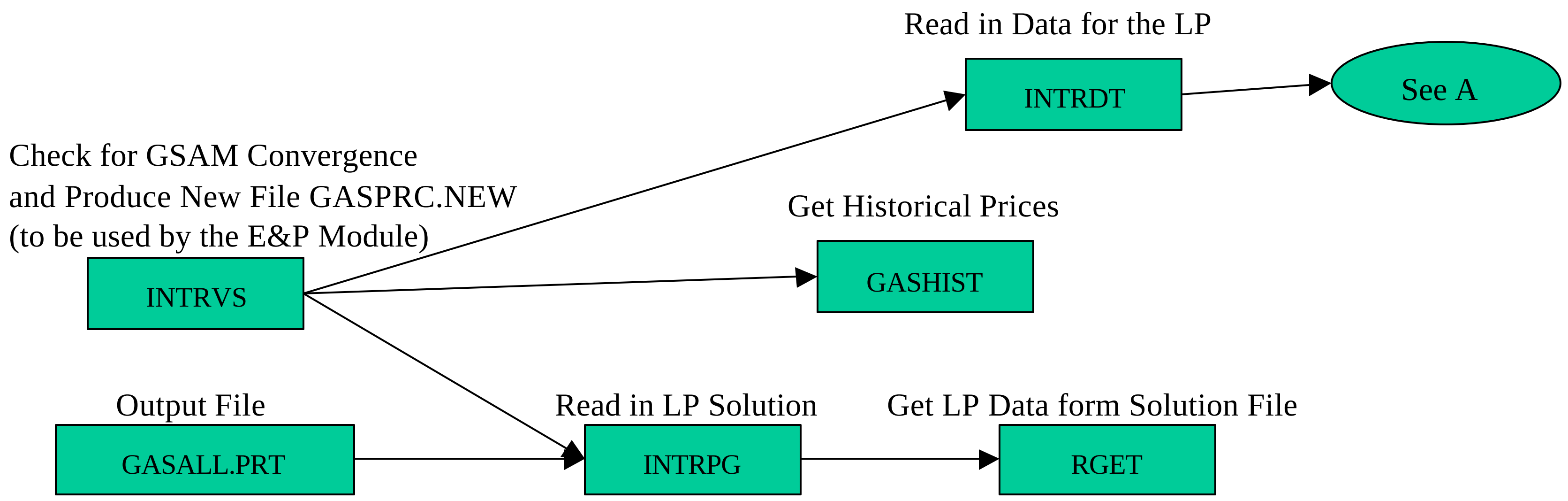
STEP 3: READ LINEAR PROGRAM SOLUTION AND PRODUCE OUTPUT REPORTS

EXECUTABLE: INTRPT.EXE

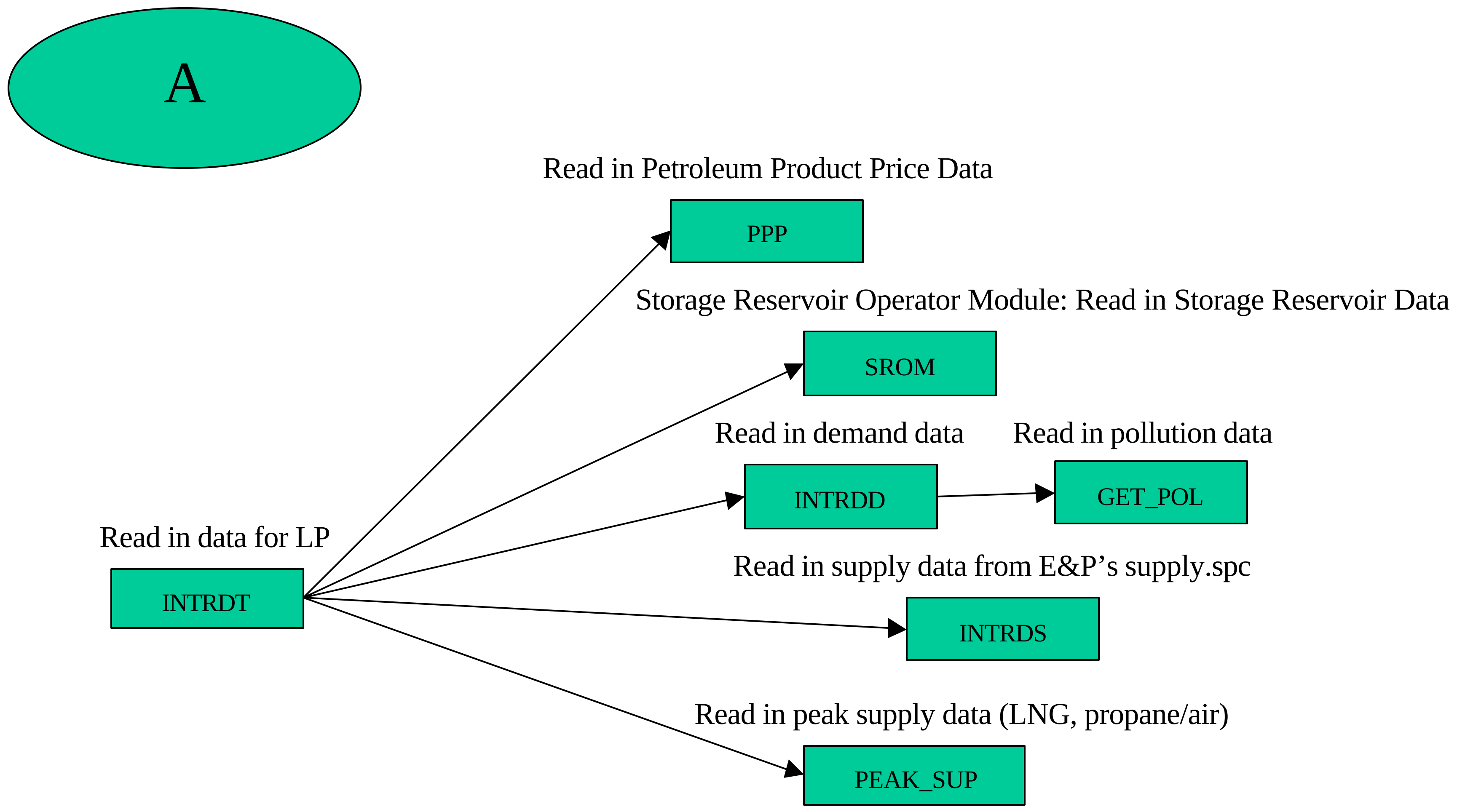

Read in Data for the LP
INTRDT
See A
Output File
GASALL.PRT
Read in LP solutions
INTRPG
Get LP Data from Solution File
RGET
Produce Transportation Report
INTRPD
Main Output File
GSAMSLN.RPT
Create Reports
INTRPT
Produce Annual Supply and Demand Report
INTRPS
Calculate Petroleum Product Prices
PPRICE2
Produce the Storage Activity Report
INTRSA
Main Output File
GSAMSLN.FLE
Produce the Storage Costs Report
INTRSC
Find First GSAM year that storage reservoirs are available
FINDFYR
Produce the Supply Summary Report
SUP_REP

STEP 4: READ LINEAR PROGRAM SOLUTION AND PRODUCE SUPPLY GAS PRICE FILE

EXECUTABLE: INTRVS.EXE



ROUTINE A


A
Read in Petroleum Product Price Data
PPP
Storage Reservoir Operator Module: Read in Storage Reservoir Data
SROM
Read in demand data
INTRDD
Read in pollution data
GET_POL
Read in data for LP
INTRDT
Read in supply data from E&P's supply.spc
INTRDS
Read in peak supply data (LNG, propane/air)
PEAK_SUP

MAIN UTILITY ROUTINES

ERRMSG - Print out error and warning messages
CHAR2NUM - convert (2 byte) character to number
GROWTH - Calculate quantities over time based on specified growth rates
NUM2CHAR - Convert number to character (2 bytes)
READ_HDR - Open file and read header lines
ZAP - Zero out an array

ALPHABETICAL LIST OF D&I SUBROUTINES

SUBROUTINE: CHAR2NUM.FOR (CHARVAL,DIGIT1,DIGIT2)

CALLED BY:	INTRPG.FOR (reads the results from the solution file.)
CALLS:	None
READS:	None
CREATES:	None
MAIN THEME:	This subroutine converts base 62 numbers (expressed as two byte characters) to the standard base 10.

Note:

Input: base 62 two digit number (charval)
(base 62 since we are using the characters,
'0',...,'9','A',...,'Z','a',...,'z')
10 + 26 + 26 = 62
Output: decimal number with digits *digit1,digit2*
such that

code(i) converts to character in '0',...,'9','A',...,'Z','a',...,'z'

charval = code(digit1)//code(digit2)

(this is useful for routine RGET.FOR which reads in the LP data)

'0',...,'9' have ASCII values of 48,...,57
'A',...,'Z' have ASCII values of 65,...,90
'a',...,'z' have ASCII values of 97,...,122

Step 1: **Define variables.**

```
v1=charval(1:1)
v2=charval(2:2)
```

Step 2: **Convert each character to a decimal number.**

```
v1num = ichar(v1)
v2num = ichar(v2)
```

Step 3: **Calculate digit1.**

```
if      (v1num.ge.48.and.v1num.le.57) then
digit1 = v1num-48        ! get decimal value
digit1 = digit1 +1      ! adjust so that code(digit1)=vcde(1:1)

else if (v1num.ge.65.and.v1num.le.90) then
digit1 = v1num-64+9     ! get decimal value
digit1 = digit1 +1      ! adjust so that code(digit1)=vcde(1:1)

else if (v1num.ge.97.and.v1num.le.122) then
digit1 = v1num-96+35    ! get decimal value
digit1 = digit1 +1      ! adjust so that code(digit1)=vcde(1:1)

else
write(*,*)'incorrect value for v1num'
stop
end if
```

Step 4: Calculate digit2.

```
if      (v2num.ge.48.and.v2num.le.57) then
digit2 = v2num-48        ! get decimal value
digit2 = digit2 +1      ! adjust so that code(digit2)=vcde(2:2)

else if (v2num.ge.65.and.v2num.le.90) then
digit2 = v2num-64+9     ! get decimal value
digit2 = digit2 +1      ! adjust so that code(digit2)=vcde(2:2)

else if (v2num.ge.97.and.v2num.le.122) then
digit2 = v2num-96+35    ! get decimal value
digit2 = digit2 +1      ! adjust so that code(digit2)=vcde(2:2)

else
write(*,*)'incorrect value for v2num'
stop
end if

return
end
```

SUBROUTINE: ERRMSG.FOR (ETYP,ECDE)

CALLED BY:	INTMGN.FOR (translates LP demand, pipeline, storage and other data into MPS format to be used as input to the LP solver) INTMGS.FOR (writes out supply information for the Integrating LP) INTRDD.FOR (reads in demand specifications from the demand input files and writes out the appropriate demand data to the LP files) INTRDS.FOR (reads in supply increment specifications) INTRPT.FOR (produces output files) INTRDT.FOR (reads in the transportation and other non-demand data for the LP) INTRVS.FOR (Checks for convergence and generates gasprc.new) PEAK_SUP (reads in the peak supply data) PPP.FOR (calculates the petroleum product prices) SROM.FOR (processes storage reservoir data)
CALLS:	None
READS:	**'**errmsg.dat' (is a file that is used to debug the program, currently blank)
CREATES:	None
MAIN THEME:	This routine prints out error and warning messages.

Note:

Arguments:

etyp is the control code and/or error type.
0 is an indicator to read in error message data.
1 is an indicator to print the informational message.
2 is an indicator to print the warning message.
3 is an indicator to print the non-fatal error message.
4 is an indicator to print the fatal error message and stop.
ecde is the message number.

Internal variables:

i is the index to message number.
k is the index to message data.
ntyp is the number of messages read in.
mtyp(i) is the message number from message data read in.
mmsg(k) is the text message.
mind(2,i) is the index to start and stop of message data for message i.
hdd(4) is the header for four types of warning message.

Step 1a: Open ‘errmsg.dat’ and test to see if message data is to be read at this step and if so read messages in. ‘errmsg.dat’ is a file used to debug the program. Close file.

```
        if(etyp.ne.0) go to 1000
        open(39,file='errmsg.dat')
        i=0

50      i=i+1
        if(i.gt.mxmsg) then
        l=001
        write(*,51) l,hdd(2)
51      format(' GSAM : ',i3,a17,'too many error messages -',
*       ' see programing support')
        i=i-1
        go to 500
        endif
        mind(1,i)=k+1
        mind(2,i)=k+1
        read(39,101,end=500) mtyp(i)
101     format(i3)
150     k=k+1
        if(k.gt.mxmsgd) then
        l=002
        write(*,51) l,hdd(2)
        k=k-1
        go to 500
        endif
        read(39,151,end=500) cdtst,mmsg(k)
151     format(a3,a70)
        mind(2,i)=k
        if(cdtst.ne.'***') go to 150
        k=k-1
        mind(2,i)=k
        go to 50

500     continue
        ntyp=i
        close(39)
        return
```

Step 1b: Find error message.

```
1000    do 1100 i=1,ntyp
        if(mtyp(i).eq.ecde) go to 1150
1100    continue
        i=0
1150    j=etyp
        if((etyp.lt.1).or.(etyp.gt.4)) j=4
```

Step 1c: Write out error message.

```
        if(i.gt.0) then
        k1=mind(1,i)
        k2=mind(2,i)
        endif
        if((i.gt.0).and.(k2.ge.k1)) then
        do 1200 k=k1,k2
        write(*,1161) ecde,hdd(j),mmsg(k)
1161    format(' GSAM : ',i3,a17,a70)
1200    continue
        else
        write(*,1201) ecde,hdd(j)
1201    format(' GSAM : ',i3,a17)
        endif
```

Step 1d: See if fatal error and stop if so, otherwise return.

```
        if(j.eq.4) then
        stop ' Fatal Error Message'
        endif
        return
        end
```

SUBROUTINE: FINDFYR.FOR (FFYR)

CALLED BY: INTRSA.FOR (writes out storage-reservoir level activities by season and year)

CALLS: None

READS: None

CREATES: None

MAIN THEME: This subroutine searches for the first GSAM year that the storage reservoir is available.

Note: ntme is the number of time periods.
strfyr is the first year that a reservoir is available.
ffyr is the first GSAM year.

Step 1: **Find the first GSAM year that the storage reservoir is available.**

```
do t=1,ntme
If(strfyr(n,t,v).gt.0) then
ffyr = t
return
end if
end do

write(*,*)'can not find ffyr in findfyr routine'
stop
return
end
```

SUBROUTINE: GASHIST.FOR

CALLED BY: INTRVS.FOR (check for convergence of equilbrium price estimates and generate ‘gasprc.new’)

CALLS: READ_HDR (open file and read the appropriate number of header lines)

READS: None

CREATES: None

MAIN THEME: This subroutine contains the gas historical routine and reads in historical gas prices (by track) starting with 1993. Note, begyr is the beginning year of the GSAM Integrating Model and is read in INTRDT.FOR.

Note: nsrg is the number of supply regions.
suppnt is the index of the supply region.
nndnme is the node name.

Historical prices can not start after 1997.

Step 1a: **Open ‘gasprc.his’ and read in data. ‘gasprc.his’ contains the historical gas prices from 1993 to (begyr-1).**

```
        Integer ii,jj,kk
        call read_hdr(31,'gasprc.his',2)
        do ii=1,5
        do s=1,nsrg
        n=suppnt(s)

        read(31,10) nndnme(n),jj,kk,
*       (hisprc(t,s,ii),t=1,(begyr-1993))
        enddo  ! s loop
        enddo  ! ii loop

10      format(a20,2i3,5(1x,f7.3))
        return
        end
```

SUBROUTINE: GET_OPT.FOR

CALLED BY: SROM.FOR (processes storage reservoir data)

CALLS: None

READS: None

CREATES: None

MAIN THEME: This subroutine calculates the costs and revenues of the storage extraction profiles, and determines which of the three storage options has the highest net profit.

Note: ldsdy is the number of days in each load segment.

Inputs:

real price(3)	! end-use regional prices from the Demand & Integrating Module ($/Mcf)
real extract(6)	! extraction rates from SRPM (%/day of working gas)
real wgas	! working gas from SRPM (Mcf)
real lc(3)	! levelized costs from SRPM ($/Mcf)
real vc(3)	! variable costs from SRPM ($/Mcf)
real profit(3)	! profit for each option ($)

Step 1: **Calculate storage deliverabilities in cubic feet per day, and net profit for each option.**

```
        deliver(1) = wgas*extract(1)/100.0   ! option 1, season 1   (5 days)
        deliver(2) = wgas*extract(2)/100.0   ! option 1, season 2   (26 days)
        deliver(3) = wgas*extract(3)/100.0   ! option 1, season 3   (90 days)

        deliver(4) = wgas*extract(4)/100.0   ! option 2, seasons 1& 2 (31 days)
        deliver(5) = wgas*extract(5)/100.0   ! option 2, season  3    (90 days)

        deliver(6) = wgas*extract(6)/100.0   ! option 3, seasons 1,2, & 3 (121 days)

        profit(1) = price(1)*ldsdy(1)*deliver(1)+
*       price(2)*ldsdy(2)*deliver(2)+
*       price(3)*ldsdy(3)*deliver(3)-
*       (lc(1)+vc(1))*(ldsdy(1)*deliver(1)+
*       ldsdy(2)*deliver(2)+ldsdy(3)*deliver(3))

        profit(2) = price(2)*(ldsdy(1)+ldsdy(2))*deliver(4)+
*       price(3)*ldsdy(3)*deliver(5)-
*       (lc(2)+vc(2))*((ldsdy(1)+ldsdy(2))*deliver(4)+
*            ldsdy(3)*deliver(5))

             profit(3) = price(3)*(ldsdy(1)+ldsdy(2)+ldsdy(3))*deliver(6)-
*            (lc(3)+vc(3))*((ldsdy(1)+ldsdy(2)+
*            ldsdy(3))*deliver(6))
```

Step 2: **Calculate option with the highest net profit. (Default option is option 1)**

```
             option  = 1
             bestval = profit(1)

             do i=2,3
             if (profit(i).gt.bestval) then
             bestval = profit(i)
             option  = i
             end if
             end do

             return
             end
```

SUBROUTINE: INTMGN.FOR

CALLED BY:	None
CALLS:	ERRMSG.FOR (prints out error and warning messages). INTMGS.FOR (writes out the supply information for the Integrating LP) INTRDT.FOR (reads in the transportation and other non-demand data for the LP). NUM2CHAR.FOR (converts base ten numbers to a 2 byte character base 62 number). PPRICE2.FOR (used in the calculation of petroleum product prices).
READS:	None
CREATES:	'gasbnd.mps'(bounds section of the LP matrix) 'gascol.mps' (columns section of the LP matrix) 'gasrhs.mps' (right hand sides section of the LP matrix) 'gasrow.mps' (rows section of the LP matrix)
MAIN THEME:	INTMGN.FOR translates LP demand, pipeline, storage and other data into MPS format to be used as input to the LP solver.

Step 1a: **Call INTRDT.FOR to read in non-gas demand related input specifications.**

```
call errmsg(0,0)
call intrdt
call errmsg(1,951)

open(27,file='prd_prc.spc')

close(27)
call errmsg(1,952)

if(nnde.gt.mxc)       call errmsg(4,301)
if(ntme.gt.mxc)       call errmsg(4,301)
if(nlds.gt.mxc)       call errmsg(4,301)
if(nlnk.gt.mxc*mxc)   call errmsg(4,301)

if(mxntad.gt.mxc)     call errmsg(4,301)
if(mxndin.gt.mxc)     call errmsg(4,301)
if(nesp.gt.mxc)       call errmsg(4,301)
if(npks.gt.mxc)       call errmsg(4,301)
```

Step 1b: Open LP MPS files and create headers for the different sections of these files.

```
        Open(11,file='gasrow.mps')
        Open(12,file='gascol.mps')
        Open(14,file='gasrhs.mps')
        Open(15,file='gasbnd.mps')

        rewind 11
100     write(11,101)
101     format('NAME',t10,'GSAM Integrating Model')
        write(11,102)
102     format('ROWS'/T3,'N',T5,'OBJ')

        rewind 12
        write(12,103)
103     format('COLUMNS')

        rewind 14
        write(14,104)
104     format('RHS')

        rewind 15
        write(15,105)
105     format('BOUNDS')
```

Step 2: Create rows specifications.

Step 2a: Set material balance constraints.

```
200     do n=1,nnde
        do t=1,ntme
        do  l=1,nlds
```

Note: Alternatively, we can use the format as specified in the line labeled ‘202’ but adjusted to read format (2x,'E',1x,'MB',3a1) to indicate that certain nodes (for example 36 and 37 as shown) should have equality material balance constraints.

```
201     format(2x,'G',1x,'MB',3a1)

        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        if ((n.eq.36).or.(n.eq.37)) then
        write(11,202) cde(n),cde(t),cde(l)
        else
        write(11,201) cde(n),cde(t),cde(l)
        end if
        end if

202     format(2x,'G',1x,'MB',3a1)
        enddo ! l loop
        enddo   ! t loop
        enddo     ! n loop
        csg
```

Step 2b: Transportation capacity constraints.

```
        do 350 q=1,nlnk
        q1=(q-1)/mxc+1
        q2=q-(q1-1)*mxc
        do 340 t=1,ntme
        do 330 l=1,nlds
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(11,301) cde(q1),cde(q2),cde(t),cde(l)
        end if

        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(16,301) cde(q1),cde(q2),cde(t),cde(l)
        end if
301     format(2x,'G',1x,'TC',4a1)
330     continue
340     continue
350     continue
```

Step 2c: Transportation capacity addition convexity constraints.

```
        do 450 q=1,nlnk
        q1=(q-1)/mxc+1
        q2=q-(q1-1)*mxc
        do 440 m=1,mxntad

        if(lnkcap(m,q).gt.0.0) then
        if((m.gt.1).or.(lnkfyr(m,q).ne.0)) then
        if(m.eq.1) then
        tcap=amax1(0.0,lnkcap(m,q))
        else
        tcap=amax1(0.0,lnkcap(m,q)-lnkcap(m-1,q))
        endif
        if(tcap.gt.0.0) then
        write(11,401) cde(q1),cde(q2),cde(m)

        write(16,401) cde(q1),cde(q2),cde(m)

401     format(2x,'L',1x,'TX',3a1)
        write(14,402) cde(q1),cde(q2),cde(m),tcap
402     format(4x,'RHS1',t15,'TX',3a1,t25,f12.4)
        endif
        endif
        endif
440     continue
450     continue
```

Step 2d: Supply convexity constraint.

```
        do 550 t=1,ntme
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(11,501) cde(t)
        end if
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(16,501) cde(t)
        end if
501     format(2x,'E',1x,'SK',a1)
550     continue
```

Step 2e: Storage volume constraints.

Note: SCntv: storage volume SVntv = sum_l Sentvl*1dsdy(l)
SDntv: storage volume SVntv = sum_l (SIntvl)*(1-strfus(n,t,v)/100)*ldsdy(l)
SXntv: storage capacity SVntv <=sum_{t'.le.t}SCnt'v
EAntv: storage extraction rate SEntv1 >= SEntv2
EBntv: storage extraction rate SEntv2 >= SEntv3

v is the index for the storage reservoir
0 .le. v .le 999 but will be represented in two digits
using the subroutine 'num2char' which takes v and gives back
a two character representation in base 62

```
        do n=1,nnde
        do t=1,ntme
        if(nsto(n,t).gt.0) then
        do v=1,nsto(n,t)
        call num2char(v,vcde)
9876    format(' v:', i3,1x,'vcde:',a2)
        if (strvcp(n,t,v).gt.0.0) then

        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(11,601) cde(n),cde(t),vcde
        end if

        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(11,602) cde(n),cde(t),vcde
        end if

        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(11,603) cde(n),cde(t),vcde
        end if

        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(11,605) cde(n),cde(t),vcde
        end if

        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(11,606) cde(n),cde(t),vcde
        end if
        endif
        enddo   ! v loop
        endif
        enddo      ! t loop
        enddo        ! n loop
        csg
```

Step 2f: Demand Convexity Constraints.

```
        do 750 r=1,ndrg
        n=dmnpnt(r)
        cmg       if (n .eq. 0) then
        cmg       endif

        do 710 t=1,ntme
```

```
          do 705 z=1,4
          do 704 f=1,7
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          write(11,701) cde(n),cde(t),cde(z),cde(f)
          end if
701       format(2x,'G',1x,'DX',4a1)
704       continue
705       continue
710       continue

          do 730 t=1,ntme
          do 725 z=1,4
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          write(11,711) cde(n),cde(t),cde(z)
          end if
          val=eutot(r,t)*eushr(r,z)/100.0
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          write(14,712) cde(n),cde(t),cde(z),val
712       format(4x,'RHS1',t15,'DY',3a1,t25,f12.4)
          end if
725       continue
730       continue
750       continue
```

Step 2g: Cost accumulation rows.

```
          do 850 t=1,ntme
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          write(11,801) cde(t)
          end if
          do 840 l=1,nlds
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          write(11,802) cde(t),cde(l)
          end if
840       continue
850       continue
```

Step 2h: Extra supply projects.

```
          do 890 e=1,nesp
          write(11,851) cde(e)
851       format(2x,'L',1x,'ES',a1)
          write(14,852) cde(e),supesq(e)
852       format(4x,'RHS1',t15,'ES',a1,t25,f12.4)
890       continue
```

Step 2i: Peak supply constraints.

Note:

indices:

p................ peak supply source (p=1 propane, p=2 LNG)
n................ node number
t................ time period
l................ gas load seasons
k................ status (k=1 existing, k=2 new)

variables:

PKOpntlk......... operating level in MMCF/day
PKIpntk.......... incremental investment level in MMCF (for year t)

constants:
pkvc(p,n,k)...... variable cost in $/MCF
pklc(p,n,k)...... levelized investment cost in $1000/MMCF/Day/year
pkfc(p,n,k)...... fixed O&M costs in $1000/MMCF
ldsdy(l)......... number of days in gas load seasons l
pkd(p,n,k)....... maximum deliverability in MMCF/day
pkfyr(p,n,k)..... first year available
pksc(p,n,k)...... storage capacity in MMCF (maximum)

constraints:
PKpntk

SUM_{l} PKOpntlk * ldsdy(l) .LE. SUM_{t' .le. t} PKIpnt'
[total operating capacity must not exceed total existing + new capacity]

PKSpnk
SUM_{t'<=t} PKIpntk .LE. pksc(p,n,k)
[total investment must not exceed total storage capacity]
bounds:
PKOpntlk .LE. pkd(p,n,k)
[operating capacity must not exceed seasonal maximum deliverability]

```
          do p=1,npks
          do r=1,ndrg
          n=dmnpnt(r)
          do k=1,2
          write(11,902) cde(p),cde(n),cde(k)
          end do
          do t=1,ntme
          do k=1,2
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          write(11,901) cde(p),cde(n),cde(t),cde(k)
          end if
          end if
          enddo ! k loop
          enddo   ! t loop
          enddo     ! r loop
          enddo       ! p loop

901       format(2x,'G',1x,'PK',4a1)
902       format(2x,'L',1x,'PKS',3a1)
```

Step 2j: Close out the rows section of the file.

```
        Endfile 11
        call errmsg(1,953)
```

Step 3: Create columns section of the MPS file.

Step 3a: Write out coefficients for forward and reverse pipeline flows.

```
1000    do 1050 q=1,nlnk
        o=lnkpnt(1,q)
        d=lnkpnt(2,q)
        do 1040 t=1,ntme
        do 1030 l=1,nlds
        val=(1.0-lnkfus(q)/100.0)
        tvom=lnkvom(q)
        q1=(q-1)/mxc+1
        q2=q-(q1-1)*mxc
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1001) cde(q1),cde(q2),cde(t),cde(l),cde(d),cde(t),
*       cde(l),val,cde(o),cde(t),cde(l),mone
        end if
1001    format(t5,'TNF',4a1,t15,'MB',3a1,t25,f12.4,
*       t40,'MB',3a1,t50,f12.4)
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1002) cde(q1),cde(q2),cde(t),cde(l),cde(q1),cde(q2),
*       cde(t),cde(l),mone,cde(t),cde(l),tvom
        end if
1002    format(t5,'TNF',4a1,t15,'TC',4a1,t25,f12.4,
*       t40,'CCD',2a1,t50,f12.4)
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1003) cde(q1),cde(q2),cde(t),cde(l),cde(o),cde(t),
*       cde(l),val,cde(d),cde(t),cde(l),mone
        end if
1003    format(t5,'TNR',4a1,t15,'MB',3a1,t25,f12.4,
*       t40,'MB',3a1,t50,f12.4)
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1004) cde(q1),cde(q2),cde(t),cde(l),cde(q1),cde(q2),
*       cde(t),cde(l),mone,cde(t),cde(l),tvom
        end if
1004    format(t5,'TNR',4a1,t15,'TC',4a1,t25,f12.4,
*       t40,'CCD',2a1,t50,f12.4)
1030    continue
1040    continue
1050    continue
```

Note: Hardwired bounds on link flows:

Canada-East	-> New England (link # 45), reverse flows should = 0
Alberta	-> WNCentral (link # 63), reverse flows should = 0
British Col.	-> Alliance-Supply (link # 74), reverse flows should = 0
Alberta	-> Alliance-Supply (link # 75), reverse flows should = 0

```
        do t=1,ntme !csg was 1,ntme
        do l=1,nlds

        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(15,1052)cde(1),cde(36),cde(t),cde(l),0.0
        write(15,1052)cde(1),cde(45),cde(t),cde(l),0.0

        write(15,1052)cde(2),cde(1),cde(t),cde(l),0.0
        write(15,1052)cde(2),cde(12),cde(t),cde(l),0.0
        write(15,1052)cde(2),cde(13),cde(t),cde(l),0.0
```

Note: Alliance has a lower bound of 1.1 Bcfd.

```
        if(t.ge.9) then
        write(15,1051)cde(2),cde(17),cde(t),cde(l),1100.0
        end if
        end if
1051    format(t2,'LO',t5,'BND1',t15,'TNF',4a1,t25,f12.4)
1052    format(t2,'UP',t5,'BND1',t15,'TNR',4a1,t25,f12.4)
        enddo   ! l loop
        enddo     ! t loop
```

Step 3b: **Write out coefficients for transportation capacity additions columns.**

```
        do 1150 q=1,nlnk
        q1=(q-1)/mxc+1
        q2=q-(q1-1)*mxc
        do 1140 t=1,ntme !csg was 1,ntme
        do 1130 m=1,mxntad


        if ((tme(t).eq.tme(ibegyr)).or.   ! first year of study
*       (m.ne.1).or.                ! new capacity
*       (lnkfyr(1,q).ne.0)) then     ! existing capacity later than 1990
        if(lnkfyr(m,q).le.tme(t)) then  ! capacity is available in year tme(t)

        if(m.eq.1) then  ! existing capacity
        tcap=lnkcap(m,q)
        else             ! new capacity
        tcap=amax1(0.0,lnkcap(m,q)-lnkcap(m-1,q))
        endif

        tccs=amax1(0.0,lnkccs(m,q))
        tfom=amax1(0.0,lnkfom(m,q))
        tcst=tccs+tfom

        if(tcap.gt.0.0) then
        if((m.gt.1).or.(lnkfyr(m,q).ne.0)) then
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then

        write(12,1101) cde(q1),cde(q2),cde(t),cde(m),cde(q1),
*       cde(q2),cde(m),one
        end if
1101    format(t5,'TA',4a1,t15,'TX',3a1,t25,f12.4)
        else
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(15,1102) cde(q1),cde(q2),cde(t),cde(m),tcap
1102    format(t2,'FX',t5,'BND1',t15,'TA',4a1,t25,f12.4)
        end if
        end if
        do 1120 t1=t,ntme  ! csg was t,ntme
        do 1110 l=1,nlds
        if(l.eq.1) then
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        if ((t1.ge.ibegyr).and.(t1.le.iendyr)) then
        write(12,1103) cde(q1),cde(q2),cde(t),cde(m),cde(q1),
*       cde(q2),cde(t1),cde(l),one,cde(t1),tcst
1103    format(t5,'TA',4a1,t15,'TC',4a1,t25,f12.4,
*       t40,'CCA',a1,t50,f12.4)
        end if
        end if
        else
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        if ((t1.ge.ibegyr).and.(t1.le.iendyr)) then
        write(12,1104) cde(q1),cde(q2),cde(t),cde(m),cde(q1),
*       cde(q2),cde(t1),cde(l),one
1104    format(t5,'TA',4a1,t15,'TC',4a1,t25,f12.4)
        end if
        end if
        end if
1110    continue
1120    continue   ! t1 loop
        endif
        endif
        endif
1130    continue
1140    continue
1150    continue
```

Step 3c: Write out coefficients for residential demand columns.

```
        vprlow=  3.30
        vprhgh= 11.00
        vinc  =  0.07

        do 1220 r=1,ndrg
        n=dmnpnt(r)
        do 1215 t=1,ntme
        tqnto=0.0
        tprco=0.0
        do 1210 j=1,((vprhgh-vprlow)/vinc)+1
        tprc=(vprhgh-vinc*float(j-1))+dmsdmr(1,r)
        tqntn=rdmbqn(t,r)*(tprc/rdmbpr(t,r))**rdmpel(r)
        tqnt=amax1((tqntn-tqnto),0.0)
        tqnto=tqntn
        call num2char(j,charval)
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(15,1201) cde(n),cde(t),charval,tqnt
1201    format(t2,'UP',t5,'BND1',t15,'DR',2a1,a2,t25,f12.4)
        end if
        do 1205 l=1,nlds
        tqntl=-1.0*(1000.0/365.0)*rdmldf(l,r)
        tprcl=tqntl*(tprc-dmsdmr(1,r))
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1202) cde(n),cde(t),charval,
*       cde(n),cde(t),cde(l),tqntl,
*       cde(t),cde(l),tprcl
1202    format(t5,'DR',2a1,a2,t15,'MB',3a1,t25,f12.4,
*       t40,'CCD',2a1,t50,f12.4)
        end if
        tqntl=-tqntl
        1205    continue
        1210   continue
        1215  continue
        1220 continue
```

Step 3d: Write out coefficients for commercial demand columns.

```
        vprlow=  2.00
        vprhgh=  9.00
        vinc  =  0.07

        do 1240 r=1,ndrg
        n=dmnpnt(r)

        do 1235 t=1,ntme
        tqnto=0.0
        tprco=0.0
        do 1230 j=1,((vprhgh-vprlow)/vinc)+1
        tprc=(vprhgh-vinc*float(j-1))+dmsdmr(2,r)
        tqntn=cdmbqn(t,r)*(tprc/cdmbpr(t,r))**cdmpel(r)
        tqnt=amax1((tqntn-tqnto),0.0)
        tqnto=tqntn
        call num2char(j,charval)
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(15,1221) cde(n),cde(t),charval,tqnt
1221    format(t2,'UP',t5,'BND1',t15,'DC',2a1,a2,t25,f12.4)
        end if
        do 1225 l=1,nlds
        tqntl=-1.0*(1000.0/365.0)*cdmldf(l,r)*(1.0-cdmish(r))
        tprcl=tqntl*(tprc-dmsdmr(2,r))
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1222) cde(n),cde(t),charval,
*       cde(n),cde(t),cde(l),tqntl,
*       cde(t),cde(l),tprcl
1222    format(t5,'DC',2a1,a2,t15,'MB',3a1,t25,f12.4,
*       t40,'CCD',2a1,t50,f12.4)
        end if
        tqntl=-tqntl

1225    continue
        if(cdmish(r).gt.0.0) then
        do 1228 l=1,nlds
        tqntl=-1.0*(1000.0/365.0)*cdmldf(l,r)*cdmish(r)
        tprcl=tqntl*(tprc-dmsdmr(2,r))
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1226) cde(n),cde(t),charval,cde(l),
*       cde(n),cde(t),cde(l),tqntl,
*       cde(t),cde(l),tprcl
1226    format(t5,'DC',2a1,a2,a1,t15,'MB',3a1,t25,f12.4,
*       t40,'CCD',2a1,t50,f12.4)
        end if
        tqntl=-tqntl
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(15,1229) cde(n),cde(t),charval,cde(l),tqnt
1229    format(t2,'UP',t5,'BND1',t15,'DC',2a1,a2,a1,t25,f12.4)
        end if
1228    continue
        endif
1230    continue
1235    continue
1240    continue
```

Step 3e: Write out coefficients for industrial demand columns.

```
        do 1300 r=1,ndrg
        n=dmnpnt(r)
        do 1295 t=1,ntme  ! csg was 1,ntme
```

Step 3f: Call PPRICE2.FOR and calculate petroleum product prices for the industrial sector.

```
        call pprice2(prccrd(t),refmar(1,t),
*       regmar(1,2,r,t),regmar(1,1,r,t),tpr(1))
        call pprice2(prccrd(t),refmar(2,t),
*       regmar(2,2,r,t),regmar(2,1,r,t),tpr(2))

        call pprice2(prccrd(t),refmar(3,t),
*       regmar(3,2,r,t),regmar(3,1,r,t),tpr(3))


        tpr(4)=99.9

        do 1290 j=1,4
        vscl=0.0

        do 1243 ss=1,niss
        tqnt=idmefc(t,r,ss)*(100.0-idmish(r,ss))/100.0*(1000.0/365.0)
        vscl=vscl+tqnt*idmshr(t,r,ss,1)*idmshr(t,r,ss,j+1)
1243    continue

        if(vscl.gt.0.0) then
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(15,1244) cde(n),cde(t),cde(j),vscl
1244    format(t2,'UP',t5,'BND1',t15,'DI',3a1,t25,f12.4)
        end if
        do 1250 l=1,nlds
        tqntl=0.0
        do 1245 ss=1,niss
        tqnt=idmefc(t,r,ss)*(1.0-idmish(r,ss)/100.0)*
*       (1000.0/365.0)
        tqntl=tqntl-tqnt*idmshr(t,r,ss,1)*idmshr(t,r,ss,j+1)*
*       idmldf(l,r,ss)
1245    continue
        if(vscl.ne.0.0) then
        tqntl=tqntl/vscl
        endif

        tprcl=amax1(0.0,tqntl*(tpr(j)-dmsdmr(3,r)))
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1246) cde(n),cde(t),cde(j),
*       cde(n),cde(t),cde(l),tqntl,
*       cde(t),cde(l),tprcl
1246    format(t5,'DI',3a1,t15,'MB',3a1,t25,f12.4,
*       t40,'CCD',2a1,t50,f12.4)
        end if
        tqntl=-tqntl

1250    continue
        endif

        do 1260 l=1,nlds
        tqntl=0.0
        do 1255 ss=1,niss
        tqnt=idmefc(t,r,ss)*idmish(r,ss)/100.0*(1000.0/365.0)
        tqntl=tqntl-tqnt*idmshr(t,r,ss,1)*idmshr(t,r,ss,j+1)*
*       idmldf(l,r,ss)
1255    continue
        vscl=-tqntl
        if(vscl.ne.0.0) then
        tqntl=tqntl/vscl
        endif
        if(tqntl.ne.0.0) then
        tprcl=tqntl*amax1(0.0,(tpr(j)-dmsdmr(3,r)))
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1256) cde(n),cde(t),cde(j),cde(l),
*       cde(n),cde(t),cde(l),tqntl,
*       cde(t),cde(l),tprcl
1256    format(t5,'DI',4a1,t15,'MB',3a1,t25,f12.4,
*       t40,'CCD',2a1,t50,f12.4)
        end if
        tqntl=-tqntl
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(15,1259) cde(n),cde(t),cde(j),cde(l),vscl
1259    format(t2,'UP',t5,'BND1',t15,'DI',4a1,t25,f12.4)
        end if
        endif
1260    continue
1290    continue
1295    continue
1300    continue
```

Step 3g: Write out coefficients for electrical utility demand columns.

```
        do 1350 r=1,ndrg
        n=dmnpnt(r)
        do 1345 t=1,ntme
        j=0
        do 1335 f=1,7
```

Step 3h: Call PPRICE2.FOR to calculate petroleum product prices in the electric utility sector.

```
        if(f.eq.1) then             ! nuclear power plants
        tprca=0.0                   ! nuclear facilities cannot burn gas

        elseif(f.eq.2) then         ! coal power plants that can burn gas
        tprca=coalpr(t,r)-dmsdmr(4,r)! price of alternative to gas is coal

        elseif(f.eq.3) then         ! hydro/other power plants
        tprca=0.0                   ! price of alternative to gas is hydro ($0$)

        elseif((f.eq.4).or.(f.eq.7)) then
        ! 4: combined cycle
        ! 7: oil/gas distillate
        ! need to calculate price of alternative to gas
        call pprice2(prccrd(t),refmar(1,t),regmar(1,2,r,t),0.0,tprca)
        tprca=tprca-dmsdmr(4,r)

        elseif(f.eq.5) then         ! oil/gas low sulfur resid
        ! need to calculate price of alternative to gas
        call pprice2(prccrd(t),refmar(2,t),regmar(2,2,r,t),0.0,tprca)
        tprca=tprca-dmsdmr(4,r)
        elseif(f.eq.6) then          ! oil/gas high sulfur resid
        ! need to calculate price of alternative to gas
        call pprice2(prccrd(t),refmar(3,t),regmar(3,2,r,t),0.0,tprca)
        tprca=tprca-dmsdmr(4,r)
        endif
        do 1330 k=1,2
        j1=j+1
        do 1316 z=1,4
        j=j+1

        if((k.eq.1).and.(tme(t).eq.begyr).and.(z.eq.1)) then !FIXES PROB.
        val=euexc(r,f,t)
        write(15,1301) cde(n),cde(t),cde(j1),val
        end if
1301    format(t2,'FX',t5,'BND1',t15,'DE',3a1,t25,f12.4)


        csg for annual model implementaion
        if((k.ne.1).or.(t.eq.ibegyr)) then
        do 1315 t1=t, iendyr ! csg was t,ntme
        if(k.eq.1) then

        if(euexc(r,f,1).gt.0.) then
        val=euexc(r,f,t1)/euexc(r,f,1)
        else
        val=1.0
        endif

        else
        val=1.0
        endif

        tprc=eunfcs(f,z,k)*1000.0*val*1000.0
        val=val*float(ndeu(z))/365.0
        if(z.eq.1) then
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        if ((t1.ge.ibegyr).and.(t1.le.iendyr)) then
        write(12,1302) cde(n),cde(t),cde(j1),
*       cde(n),cde(t1),cde(z),cde(f),val,
*       cde(t1),tprc
        if ((t.eq.2).and.(t1.eq.2)) then
        write(22,1302) cde(n),cde(t),cde(j1),
*       cde(n),cde(t1),cde(z),cde(f),val,
*       cde(t1),tprc
        end if

1302    format(t5,'DE',3a1,t15,'DX',4a1,t25,f12.4,
*       t40,'CCA',a1,t50,f12.4)
        end if
        end if
        else
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        if ((t1.ge.ibegyr).and.(t1.le.iendyr)) then
        write(12,1303) cde(n),cde(t),cde(j1),
*       cde(n),cde(t1),cde(z),cde(f),val

        if ((t.eq.2).and.(t1.eq.2)) then
        write(22,1303) cde(n),cde(t),cde(j1),
*       cde(n),cde(t1),cde(z),cde(f),val
        end if
```

```
1303    format(t5,'DE',3a1,t15,'DX',4a1,t25,f12.4)
        end if
        endif
        end if
1315    continue     ! t1 loop
        endif
1316    continue       ! z loop
        j=j1-1
        do 1329 z=1,4
        j=j+1
        if((f.ge.1).and.(f.le.3)) then   ! nuclear, coal, hydro (respectively)
        l2=nlds+1
        vscl=0.0
        else
        l2=1
        vscl=1.0
        endif

        if(k.eq.1) then
        do 1325 l1=l2,nlds+1

1399    format(t2,'FX',t5,'BND1',t15,'DE',4a1,t25,f12.4)


        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1317) cde(n),cde(t),cde(j),cde(l1),
*       cde(n),cde(t),cde(z),one,
*       cde(n),cde(t),cde(z),cde(f),mone
1317    format(t5,'DE',4a1,t15,'DY',3a1,t25,f12.4,
*       t40,'DX',4a1,t50,f12.4)
        end if
        do 1320 l=1,nlds
        if(l.lt.l1) then

        tqntl=euldf(l,r,z)*1000.0*1000.0/365.0
        tprcl=(eusox(f)+euoenv(f)+tprca*euflef(f,z,k)/1000.0+
1       eunvcs(f,z,k))*tqntl


        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1318) cde(n),cde(t),cde(j),cde(l1),
*       cde(t),cde(l),tprcl
1318    format(t5,'DE',4a1,t15,'CCD',2a1,t25,f12.4)
        end if
        else

        tqntl=-euflef(f,z,k)*euldf(l,r,z)*(1000.0/365.0)


        tprcl=-eunvcs(f,z,k)*tqntl

        tprcl=eunvcs(f,z,k)*euldf(l,r,z)*1000.0*1000.0/365
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1319) cde(n),cde(t),cde(j),cde(l1),
*       cde(n),cde(t),cde(l),tqntl,
*       cde(t),cde(l),tprcl
1319    format(t5,'DE',4a1,t15,'MB',3a1,t25,f12.4,
*       t40,'CCD',2a1,t50,f12.4)
        end if
        endif
1320    continue      ! l loop
1325    continue       ! l1 loop
```

```
        endif
1329    continue       ! z loop
1330    continue        ! k loop
1335    continue         ! f loop
1345    continue          ! t loop
1350    continue           ! r loop
```

Step 3i: Write out coefficients for storage volume columns.

```
        do n=1,nnde
        do t=1,ntme  !csg was 1,ntme
        if(nsto(n,t).gt.0) then
        do v=1,nsto(n,t)
        call num2char(v,vcde)
        if (strvcp(n,t,v).gt.0.0) then

        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1401) cde(n),cde(t),vcde,
1       cde(n),cde(t),vcde,mone
1401    format(t5,'SV',2a1,a2,t15,'SX',2a1,a2,t25,f12.4)
        end if
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(15,1402) cde(n),cde(t),vcde,strvcp(n,t,v)
1402    format(t2,'UP',t5,'BND1',t15,'SV',2a1,a2,t25,f12.4)
        end if
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1403)cde(n),cde(t),vcde,
1       cde(n),cde(t),vcde,one,
2       cde(n),cde(t),vcde,one
1403    format(t5,'SV',2a1,a2,t15,'SC',2a1,a2,t25,f12.4,
*       t40,'SD',2a1,a2,t50,f12.4)
        end if
        endif
        enddo  ! v loop
        endif
        enddo       ! t loop
        enddo        ! n loop
```

Step 3j: Write out coefficients for storage capacity additions columns.

```
        do n=1,nnde
        do t=1,ntme !csg was 1,ntme
        if(nsto(n,t).gt.0) then
        do v=1,nsto(n,t)
        call num2char(v,vcde)
        if (strvcp(n,t,v).gt.0.0) then

        if ((t.ge.ibegyr).and.(t.le.iendyr).and.
*       (strfyr(n,t,v).eq.1))  then
        if (t.eq.ibegyr) then
        write(15,1500) cde(n),cde(t),vcde,strvcp(n,t,v)
        else
        write(15,1500) cde(n),cde(t),vcde,0.0
        endif
        endif

1500    format(t2,'FX',t5,'BND1',t15,'SC',2a1,a2,t25,f12.4)

        do t1=t,iendyr ! csg was t,ntme
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1501) cde(n),cde(t),vcde,
1       cde(n),cde(t1),vcde,one
1501    format(t5,'SC',2a1,a2,t15,'SX',2a1,a2,t25,f12.4)
        end if

        tcst=strcst(n,t,v)+strfom(n,t,v)
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1502) cde(n),cde(t),vcde,
1       cde(t1),tcst
1502    format(t5,'SC',2a1,a2,t15,'CCA',a1,t25,f12.4)
        end if
        enddo    ! t1 loop
        endif
        enddo       ! v loop
        endif
        enddo            ! t loop
        enddo              ! n loop
```

Step 3k: Write out coefficients for storage extraction columns.

```
        do n=1,nnde
        do t=1,ntme  !csg was 1,ntme
        if(nsto(n,t).gt.0) then
        do v=1,nsto(n,t)
        call num2char(v,vcde)
        if (strvcp(n,t,v).gt.0.0) then

        do l=1,nlds
        val=-float(ldsdy(l))

        csg put bounds on the storage extraction amounts for each load
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(15,1600)cde(n),cde(t),vcde,cde(l),
1       strvcp(n,t,v)*strecp(n,t,v,l)/100.0
1600    format(t2,'UP',t5,'BND1',t15,'SE',2a1,a2,a1,
1       t25,f12.4)
        end if
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1601) cde(n),cde(t),vcde,cde(l),
1       cde(n),cde(t),cde(l),one,
2       cde(n),cde(t),vcde,val
        end if

1601    format(t5,'SE',2a1,a2,a1,t15,'MB',3a1,t25,f12.4,
1       t40,'SC',2a1,a2,t50,f12.4)


        if (l.eq.1) then
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1603) cde(n),cde(t),vcde,cde(l),
1       cde(n),cde(t),vcde,one
        end if
        else if (l.eq.2) then
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1604) cde(n),cde(t),vcde,cde(l),
1       cde(n),cde(t),vcde,mone,
2       cde(n),cde(t),vcde,one
        end if
        else if (l.eq.3) then
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1605) cde(n),cde(t),vcde,cde(l),
1       cde(n),cde(t),vcde,mone
        end if
        else
        end if
1603    format(t5,'SE',2a1,a2,a1,t15,'EA',2a1,a2,t25,f12.4)
1604    format(t5,'SE',2a1,a2,a1,t15,'EA',2a1,a2,t25,f12.4,
1       t40,'EB',2a1,a2,t50,f12.4)
1605    format(t5,'SE',2a1,a2,a1,t15,'EB',2a1,a2,t25,f12.4)

        enddo   ! l loop
        endif
        enddo       ! v loop
        endif
        enddo             ! t loop
        enddo               ! n loop
        csg
```

Step 3l: Write out coefficients for storage injection columns.

```
        do n=1,nnde
        do t=1,ntme  !csg was 1,ntme
        if(nsto(n,t).gt.0) then
        do v=1,nsto(n,t)
        call num2char(v,vcde)
        if (strvcp(n,t,v).gt.0.0) then
        do l=1,nlds
        val=-float(ldsdy(l))*(1.0-strfus(n,t,v)/100.0)

        csg put bounds on the storage injection amounts for each load
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(15,1700)cde(n),cde(t),vcde,cde(l),
1       strvcp(n,t,v)*stricp(n,t,v,l)/100.0
1700    format(t2,'UP',t5,'BND1',t15,'SI',2a1,a2,a1,
1       t25,f12.4)
        end if
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1701) cde(n),cde(t),vcde,cde(l),
1       cde(n),cde(t),cde(l),mone,
2       cde(n),cde(t),vcde,val
1701    format(t5,'SI',2a1,a2,a1,t15,'MB',3a1,t25,f12.4,
1       t40,'SD',2a1,a2,t50,f12.4)
        end if
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1702) cde(n),cde(t),vcde,cde(l),
1       cde(t),cde(l),strvom(n,t,v)
1702    format(t5,'SI',2a1,a2,a1,t15,'CCD',2a1,t25,f12.4)
        end if
        enddo  ! l loop
        endif
        enddo      ! v loop
        endif
        enddo            ! t loop
        enddo              ! n loop
```

Step 3m: Write out coefficients for extra supply columns.

```
        do 1850 e=1,nesp
        do 1840 t=1,ntme ! csg was 1,ntme
        if(supesy(e).le.tme(t)) then
        n=supesn(e)
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,1801) cde(e),cde(t),cde(e),one
1801    format(t5,'ES',2a1,t15,'ES',a1,t25,f12.4)
        end if
        do 1830 t1=t,ntme ! csg was t,ntme
        do 1820 l=1,nlds
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        if ((t1.ge.ibegyr).and.(t1.le.iendyr)) then
        write(12,1802) cde(e),cde(t),cde(n),cde(t1),cde(l),one,
*       cde(t1),cde(l),supesp(e)
1802    format(t5,'ES',2a1,t15,'MB',3a1,t25,f12.4,
*       t40,'CCD',2a1,t50,f12.4)
        end if
        end if
1820    continue
1830    continue
        endif
1840    continue
1850    continue
```

Step 3n: **Write out coefficients for operating peak supply columns.**

Note: Write out coefficients for constraints concerning:

material balance……………. (MBntl)
peak supply investment……..(PKpntk)
daily costs…………………...(CCDtl)

```
        do p=1,npks
        do r=1,ndrg
        n=dmnpnt(r)
        do t=1,ntme ! csg was 1,ntme
        do l=1,nlds
        do k=1,2
        val = float(ldsdy(l))
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        if (pkfyr(p,n,k).le.tme(t)) then
        write(12,1901) cde(p),cde(n),cde(t),cde(l),cde(k),
*       cde(n),cde(t),cde(l),one,
*       cde(p),cde(n),cde(t),cde(k),-val
1901    format(t5, 'PKO',5a1,t15,'MB', 3a1,t25,f12.4,
*        t40,'PK', 4a1,t50,f12.4)

        write(12,1902) cde(p),cde(n),cde(t),cde(l),cde(k),
*       cde(t),cde(l),pkvc(p,n,k)
1902    format(t5,'PKO',5a1,t15,'CCD',2a1,t25,f12.4)
        endif  ! pkfyr clause
        endif  ! t clause

        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        if (pkfyr(p,n,k).le.tme(t)) then
        write(15,1903)cde(p),cde(n),cde(t),cde(l),cde(k),
*       pkd(p,n,k)
1903    format(t2,'UP',t5,'BND1',t15,'PKO',5a1,t25,f12.4)

        if ((l.eq.1).and.(k.eq.1)) then
        write(15,1906)cde(p),cde(n),cde(t),cde(l),cde(k),
*       pklbnd(p,n)
        endif
```

```
1906    format(t2,'LO',t5,'BND1',t15,'PKO',5a1,t25,f12.4)

        if (l.ne.1) then
        write(15,1899)cde(p),cde(n),cde(t),cde(l),cde(k),
*       0.0
1899    format(t2,'FX',t5,'BND1',t15,'PKO',5a1,t25,f12.4)
        end if
        endif   ! pkfyr clause
        endif   ! t clause
        enddo ! k loop
        enddo   ! l loop
        enddo      ! t loop
        enddo        ! r loop
        enddo          ! p loop


        call errmsg(1,954)
```

Step 3o: Write out coefficients for investment capacity in peak supply columns.

Note: Write out coefficients for constraints concerning:

peak supply investment..........	(PKpntk)
peak supply maximum investment..	(PKSpnk)
annual costs....................	(CCAt)

```
        do p=1,npks
        do r=1,ndrg
        n=dmnpnt(r)
        do k=1,2


        write(14,1904) cde(p),cde(n),cde(k),pksc(p,n,k)
1904    format(4x,'RHS1',t15,'PKS',3a1,t25,f12.4)
        end do

        do t=1,ntme
        do k=1,2
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        if (pkfyr(p,n,k).le.tme(t)) then
        write(12,1905) cde(p),cde(n),cde(t),cde(k),
*       cde(p),cde(n),cde(k),one
1905    format(t5,'PKI',4a1,t15,'PKS',3a1,t25,f12.4)
        endif ! pkfyr clause
        endif ! t clause

        do t1=t,iendyr !csg was t,ntme
        val = pklc(p,n,k)/365.+pkfc(p,n,k)
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        if (pkfyr(p,n,k).le.tme(t)) then
        write(12,1911) cde(p),cde(n),cde(t),cde(k),
*       cde(p),cde(n),cde(t1),cde(k),one,
*       cde(t1),val
1911    format(t5,'PKI',4a1,t15,'PK',4a1,t25,f12.4,
*       t40,'CCA',a1,t50,f12.4)

        endif ! pkfyr clause
        endif ! t clause
        enddo ! t1 loop
        enddo    ! k loop
        enddo      ! t loop
        enddo        ! r loop
        enddo          ! p loop

        call errmsg(1,954)
```

Step 4a: Call INTMGS.FOR to create supply vectors.

```
        call intmgs
        call errmsg(1,955)
```

Step 5a: Write out vectors that convert costs to present value.

```
        vm1000=-1000.
        do 2050 t=1,ntme
        vald=(1.0/(1.0+disrte/100.))**(1./365.)
        vala=(1.0/(1.0+disrte/100.))
        vals=0.0
        valt=0.0

        do 2040 l=1,nlds
        val=vald**vals*(vald**ldsdy(l)-1.0)/(vald-1.0)
        val=val*vala**(tme(t)-tme(ibegyr))
        valt=valt+val
        vals=vals+ldsdy(l)
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,2001) cde(t),cde(l),cde(t),cde(l),vm1000,val
2001    format(t5,'OD',2a1,t15,'CCD',2a1,t25,f12.4,t40,'OBJ',
*       t50,f10.4)
        end if
        val=0.0
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(15,2002) cde(t),cde(l),val
2002    format(t2,'FR',t5,'BND1',t15,'OD',2a1,t25,f12.4)
        end if
2040    continue

        valt=valt/365.0
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(12,2041) cde(t),cde(t),vm1000,valt
2041    format(t5,'OA',a1,t15,'CCA',a1,t25,f12.4,t40,'OBJ',t50,f10.4)
        end if
2050    continue
```

Step 5b: Put end-of-matrix flag at end of bounds section and terminate.

```
        call errmsg(1,956)
        endfile 12
        endfile 14
        endfile 15
        stop
        end
```

SUBROUTINE: INTMGS.FOR

CALLED BY: INTMGN.FOR (translates LP demand, pipeline, storage and other data into MPS format to be used as input to the LP solver)

CALLS: ERRMSG.FOR (prints out error and warning messages)

READS: None

CREATES: ‘gascls.mps’ (supply columns section of the LP matrix)
‘gasbns.mps’ (supply bounds section of the LP matrix)
‘gasrss.mps’ (supply right-hand side section of the LP matrix)

MAIN THEME: This subroutine writes out supply information for the Integrating LP.

Note: mxnlds is the maxinum number of load segments.
mxntme is the maxinum number of time periods.
vscl is a scaling factor.
nsup is the number of supply increments.

Step 1a: **Open output files, ‘gascls.mps’, ‘gasbns.mps’, ‘gasrss.mps’, (these are parts of ‘gasall.mps’, the major input file to the integrating LP).**

```
        Open(13,file='gascls.mps')
        Open(16,file='gasbns.mps')
        Open(29,file='gasrss.mps')

        rewind 13
        rewind 16
```

Step 1b: **Write supply data to MPS files for the LP.**

```
        if(nsup.gt.mxc) call errmsg(4,301)
        vscl=0.0
        nscl=0
        do 90 k=1,nsup
        ulmt=max(0.0,1.0-pscale*(nsps-suppas(k)))
        if(ulmt.gt.0.0) then
        do 80 s=1,nsrg
        do 70 t=1,ntme
        if(suptot(t,s,k).gt.0.0) then
        vscl=vscl+suptot(t,s,k)*(1.0-lsepln/100.0)
        nscl=nscl+1
        endif
70      continue
80      continue
        endif
```

```
90        continue
          if((nscl.gt.1).and.(vscl.ne.0.0)) then
          vscl=vscl/float(nscl)
          else
          vscl=1.0
          endif

          do 95 t=1,ntme
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          write(29,92) cde(t),vscl
          end if
92        format(4x,'RHS1',t15,'SK',a1,t25,f12.4)
95        continue

100       do 195 t1=1,ntme
          if(t1.eq.ibegyr) then
          vleft=1.0
          else
          vleft=1.0-vdr**(tme(t1)-tme(t1-1))
          endif

          do 190 k=1,nsup
          ulmt=max(0.0,1.0-pscale*(nsps-suppas(k)))

          if(ulmt.gt.0.0) then
          if ((t1.ge.ibegyr).and.(t1.le.iendyr)) then
          write(13,101) cde(k),cde(t1),cde(t1),one
101       format(t5,'SP',2a1,t15,'SK',a1,t25,f12.4)
          end if
          if(ulmt.lt.1.0) then
          valq=ulmt*vscl
          if ((t1.ge.ibegyr).and.(t1.le.iendyr)) then
          write(16,102) cde(k),cde(t1),valq
102       format(t2,'UP',t5,'BND1',t15,'SP',2a1,t25,f12.4)
          end if
          endif
          do 120 t=1,ntme
          do 115 l=1,nlds
          valp(l,t)=0.0
115       continue
120       continue

          do 180 s=1,nsrg
          n=suppnt(s)
          il=0
          do 170 t=t1,ntme
          vdo=vleft*vdr**(tme(t)-tme(t1))
          do 160 l=1,nlds
          valq=supldf(s,l)*suptot(t,s,k)/vscl*(1.0-lsepln/100.0)*vdo
          valp(l,t)=valp(l,t)+valq*(supavp(t,s,k)+gthcst(s))
          if(il.eq.0) then
          valqo=valq
          il=l
          it=t
          else
          vlqo=valqo
          lq=valq
           if ((t1.ge.ibegyr).and.(t1.le.iendyr)) then
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          write(13,131) cde(k),cde(t1),cde(n),cde(it),cde(il),vlqo,
*         cde(n),cde(t),cde(l),vlq
131       format(t5,'SP',2a1,t15,'MB',3a1,t25,f12.6,
*       t40,'MB',3a1,t50,f12.6)
          end if
          end if
          il=0
          endif
160     continue
170     continue
          if(il.ne.0) then
          vlqo=valqo
          if ((t1.ge.ibegyr).and.(t1.le.iendyr)) then
          if ((it.ge.ibegyr).and.(it.le.iendyr)) then
          write(13,171) cde(k),cde(t1),cde(n),cde(it),cde(il),valqo
171     format(t5,'SP',2a1,t15,'MB',3a1,t25,f12.6)
          end if
          end if
          il=0
          end if
180     continue
          il=0
          do 185 t=1,ntme
          do 184 l=1,nlds
          if(il.eq.0) then
          il=l
          it=t
          else
          if ((t1.ge.ibegyr).and.(t1.le.iendyr)) then
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          write(13,181) cde(k),cde(t1),cde(it),cde(il),valp(il,it),
*       cde(t),cde(l),valp(l,t)
181     format(t5,'SP',2a1,t15,'CCD',2a1,t25,f12.6,t40,'CCD',2a1,
*       t50,f12.6)
          end if
          end if
          il=0
          endif
184     continue
185     continue
          if(il.ne.0) then
          if ((t1.ge.ibegyr).and.(t1.le.iendyr)) then
          if ((it.ge.ibegyr).and.(it.le.iendyr)) then
          write(13,186) cde(k),cde(t1),cde(it),cde(il),valp(il,it)
186     format(t5,'SP',2a1,t15,'CCD',2a1,t25,f12.6)
          end if
          end if
          endif
          endif
190     continue
195     continue
          write(16,201)
201     format('ENDATA')
```

Step 1c: Put end of file markers and return.

```
        endfile 13
        endfile 16
        return
        end
```

SUBROUTINE: INTRDD.FOR

CALLED BY:	INTRDT.FOR (reads in the transportation and other non-demand data for the LP)
CALLS:	ERRMSG.FOR (prints out error and warning messages) GROWTH (calculates the rates over time with growth factors read in from a file) READ_HDR (reads in the appropriate number of header lines in a file)
READS:	**‘**com_dem.spc’ (contains commmercial demand values) ‘com_eff.spc’ (contains commmercial energy efficiency values) ‘com_els.spc’ (contains commercial elasticity values) ‘com_gnp.spc’ (contains commecial GNP values) ‘com_ld.spc’ (contains commecial load factors) ‘com_prc.spc’ (contains commercial prices) ‘eu_dem.spc’ (contains electricity demand figures) ‘eu_gen.spc’ (contains the electric utility cost and efficiency data by plant type) ‘eu1_ld.spc’ (contains the electric utility load factors for season 1) ‘eu2_ld.spc’ (contains the electric utility load factors for season 2) ‘eu3_ld.spc’ (contains the electric utility load factors for season 3) ‘eu4_ld.spc’ (contains the electric utility load factors for season 4) ‘ind_ld.spc’ (contains the industrial demand load factors) ‘ind_dem.spc’ (contains the industrial demand values) ‘pop_grp.spc’ (contains data on population, economic, and electricity demand growth) ‘res_eff.spc’ (contains residential energy efficiency values) ‘res_els.spc’ (contains residential elasticity values) ‘res_ld.spc’ (contains residential load factor values) ‘res_pop.spc’ (contains residential population figures) ‘res_prc.spc’ (contains residential prices)
CREATES:	None
MAIN THEME:	This routine reads in demand specifications from the demand input files and prepares the variables that will be written as inputs to the linear program.

Note:

tt is a counter for ‘pop_grp.spc’
jj is an index of a data type
ndrg is the number of demand regions.

nndnme is the node name
econspc is the economic specifications array
popspc is the population's specifications array
elecspc is the electricity demand specifications array

Step 1a: **Open the input file 'pop_grp.spc' which contains data on population, gross regional product, electricity demand growth. READ_HDR is a subroutine that opens a file and reads in the appropriate number of header lines. Read in gross regional product, economic, and electricity demand growth assumptions. Close file.**

```
        call read_hdr(21,'pop_grp.spc',7)
        do 90 r=1,ndrg
        iflpe(1,r)=0
        iflpe(2,r)=0
90      continue
        ifl=0
        tt=0
```

Step 1b: **Determine which data type is currently being read in:**
jj=1® population;
jj=2® gross regional product;
jj=3® electricity demand growth.

```
100     tt=tt+1
        if(tt.le.ndrg) then
        jj=1
        elseif(tt.eq.(ndrg+1)) then
        read(21,*)
        goto 100
        elseif(tt.le.(ndrg+ndrg+1)) then
        jj=2
        elseif(tt.eq.(ndrg+ndrg+2)) then
        read(21,*)
        goto 100
        elseif(tt.eq.(ndrg+ndrg+3)) then
        jj=3
        else
        goto 150
        endif
```

Step 1c: **Call subroutine GROWTH to calculate a forecast for each data type (population, gross regional product, electricity demand).**

```
        read(21,*,end=150) nname,tbse,(trates(t),t=1,5)
        do 105 t=1,ntme
        call growth(tbse,trates,t,tval)

        tinp(t)=tval
105     continue

        if((jj.eq.3).and.(nname.eq.'TOTAL          ')) go to 115
```

```
          do 110 r=1,ndrg
          n=dmnpnt(r)
          if((jj.lt.3).and.(nndnme(n).eq.nname)) go to 115
110       continue
          write(*,111) nname
111       format(' D-Region: ',a20)
          call errmsg(4,501)
```

Step 1d: **Store the values for population, gross regional product, and electricity demand in the appropriate arrays.**

```
115       if(jj.lt.3) then
          if(iflpe(jj,r).ne.0) then
          call errmsg(4,501)
          endif
          iflpe(jj,r)=1
          else
          if(ifl.ne.0) then
          call errmsg(4,501)
          endif
          ifl=0
          endif
          do 120 t=1,ntme
          if(jj.eq.1) then
          econsp(t,r)=tinp(t)
          elseif(jj.eq.2) then
          popspc(t,r)=tinp(t)
          else
          elecsp(t)=tinp(t)
          endif
120       continue
          go to 100

150       close(21)
          do 160 r=1,ndrg
          do 155 jj=1,2
          if(iflpe(jj,r).ne.1) call errmsg(4,502)
155       continue
160       continue
```

Note:

bsedem is base demand.
demrate is demand growth rate.
prcels is price elasticity.
popels is population elasticity.
bseeff is base efficiency.
effrates is efficiency growth rates.
bseprc is base price.
prcrate is the growth rate of price.
bsepop is base population.
poprates is population growth rate.

Step 2a: **Read in the residential demand data. Open files and read header lines. The residential demand data is read from the following files:**

'res_dem.spc' (contains residential demand values);
'res_ld.spc' (contains residential load factor values);
'res_els.spc' (contains residential elasticity values);
'res_eff.spc' (contains residential energy efficiency values);
'res_prc.spc' (contains residential prices);
'res_pop.spc' (contains residential population).

```
        call read_hdr(11,'res_dem.spc',5)
        call read_hdr(12,'res_ld.spc', 3)
        call read_hdr(13,'res_els.spc',3)
        call read_hdr(14,'res_eff.spc',5)
        call read_hdr(15,'res_prc.spc',5)
        call read_hdr(16,'res_pop.spc',5)
```

Step 2b: **Initialize load factors to zero.**

```
        do l=nlds+1,mxnlds
        linp(l) = 0.0
        enddo
```

Note: iflr(r) is a read-in flag to make sure each region is read in (0 means region r not yet read, 1 otherwise)

```
        do r=1,ndrg
        iflr(r)=0
        enddo
```

Step 2c: **Read data from all residential files, one region at a time.**

```
220     read(11,*,end=260) nname1,bsedem ,(demrates(t),t=1,5)
        read(12,*)         nname2,(linp(l),l=1,nlds)
        read(13,*)         nname3,prcels,popels
        read(14,*)         nname4,bseeff,(effrates(t),t=1,5)
        read(15,*)         nname5,bseprc,(prcrates(t),t=1,5)
        read(16,*)         nname6,bsepop,(poprates(t),t=1,5)
```

Step 2d: **Check to make sure region (node) names are consistent between the files.**

```
        if ((nname1.eq.nname2).and.(nname2.eq.nname3).and.
1       (nname3.eq.nname4).and.(nname4.eq.nname5).and.
2       (nname5.eq.nname6)) go to 230
        write(*,*) 'region names/region order are inconsistent'
        write(*,*) 'in the residential sector files'
        write(*,*) 'program aborted'
        stop

230     continue
```

Step 2e: Check for incorrect elasticity values.

```
        if((prcels.gt.0.0).or.(popels.lt.0.0)) then
        write(*,222) nname
222     format(' Demand Region: ',a20)
        call errmsg(4,505)
        endif
```

Step 2f: Find the pointer to the current node name.

```
        do r=1,ndrg
        n=dmnpnt(r)
        if(nndnme(n).eq.nname1) go to 235
        enddo

        write(*,222) nname
        call errmsg(4,506)

235
        if(iflr(r).ne.0) then
        write(*,222) nname
        call errmsg(4,507)
        endif
```

Step 2g: Set read-in flag =1 (regional data read)

```
        iflr(r)=1
        rdmpel(r)=prcels
```

Step 2h: Calculate residential, population, and efficiency figures for each year.

```
        do yr=1,ntme
        call growth(bsedem,demrates,yr,dem_val)
        call growth(bseeff,effrates,yr,eff_val)
        call growth(bseprc,prcrates,yr,prc_val)
        call growth(bsepop,poprates,yr,pop_val)
        pop_val=(pop_val/popspc(yr,r))**popels   ! normalize and exponentiate
        rdmbqn(yr,r) = dem_val*pop_val*eff_val
        rdmbpr(yr,r)=prc_val
        enddo  ! yr loop

        vscl=0.0
        do l=1,nlds
        linp(l)=amax1(linp(l),0.0)
        vscl=vscl+linp(l)*ldsdy(l)/365.0
        enddo

        do l=1,nlds
        if(vscl.le.0.0) then
        rdmldf(l,r)=1.0
        else
        rdmldf(l,r)=linp(l)/vscl
        endif
        enddo
        go to 220
```

Step 2i: **Close residential demand files.**

```
260     close (11)
        close (12)
        close (13)
        close (14)
        close (15)

        do r=1,ndrg
        if(iflr(r).eq.0) then
        n=dmnpnt(r)
        write(*,222) nndnme(n)
        call errmsg(4,508)
        endif
        enddo
```

Note: bsegnp is the base GNP.
gnprates is the GNP growth rate.
gnpels is the GNP elasticity.

Step 3a: **Read in commercial demand data. Open files and read header lines. The commercial demand data is read in from the following files:**

'com_dem.spc' (contains commercial demand values);
'com_ld.spc' (contains commercial load factors);
'com_els.spc' (contains commercial elasticity values);
'com_eff.spc' (contains commercial energy efficiency values);
'com_prc.spc' (contains commercial prices);
'com_gnp.spc' (contains commercial GNP values).

```
        call read_hdr(11,'com_dem.spc',5)
        call read_hdr(12,'com_ld.spc', 3)
        call read_hdr(13,'com_els.spc',3)
        call read_hdr(14,'com_eff.spc',5)
        call read_hdr(15,'com_prc.spc',5)
        call read_hdr(16,'com_gnp.spc',5)
```

Step 3b: **Initialize load factors to zero.**

```
        do l=nlds+1,mxnlds
        linp(l) = 0.0
        enddo

        do r=1,ndrg
        iflr(r)=0
        enddo
```

Step 3c: **Read data from all commercial files, one region at a time.**

```
320     read(11,*,end=360) nname1,bsedem ,(demrates(t),t=1,5),ishr
        read(12,*)       nname2,(linp(l),l=1,nlds)
        read(13,*)       nname3,prcels,gnpels
        read(14,*)       nname4,bseeff,(effrates(t),t=1,5)
        read(15,*)      nname5,bseprc,(prcrates(t),t=1,5)
        read(16,*)      nname6,bsegnp,(gnprates(t),t=1,5)
```

Step 3d: Check to make sure region (node) names are consistent between the files.

```
        if ((nname1.eq.nname2).and.(nname2.eq.nname3).and.
1       (nname3.eq.nname4).and.(nname4.eq.nname5).and.
2       (nname5.eq.nname6)) go to 330
        write(*,*) 'region names/region order are inconsistent'
        write(*,*) 'in the commercial sector files'
        write(*,*) 'program aborted'
        stop

330     continue
        if((prcels.gt.0.0).or.(popels.lt.0.0)) then
        write(*,222) nname
        call errmsg(4,505)
        endif
```

Step 3e: Find the pointer to the current node name.

```
        do r=1,ndrg
        n=dmnpnt(r)
        if(nndnme(n).eq.nname1) go to 335
        enddo

        write(*,222) nname
        call errmsg(4,506)

335     if(iflr(r).ne.0) then
        write(*,222) nname
        call errmsg(4,507)
        endif
```

Step 3f: Set read-in flag =1 (regional data used).

```
        iflr(r)=1
        cdmpel(r)=prcels
        cdmish(r)=ishr/100.0
```

Step 3g: Calculate the commercial GRP and efficiency figures for each year.

```
        do yr=1,ntme
        call growth(bsedem,demrates,yr,dem_val)
        call growth(bseeff,effrates,yr,eff_val)
        call growth(bseprc,prcrates,yr,prc_val)
        call growth(bsegnp,gnprates,yr,gnp_val)
        gnp_val=(gnp_val/econsp(yr,r))**gnpels
        cdmbqn(yr,r) = dem_val*gnp_val*eff_val
        cdmbpr(yr,r)=prc_val
        enddo  ! yr loop

        vscl=0.0
        do l=1,nlds
        linp(l)=amax1(linp(l),0.0)
        vscl=vscl+linp(l)*ldsdy(l)/365
        enddo

        do l=1,nlds
        if(vscl.le.0.0) then
        cdmldf(l,r)=1.0
        else
        cdmldf(l,r)=linp(l)/vscl
        endif
        enddo
        go to 320
```

Step 3h: Close commercial demand files.

```
360     close (11)
        close (12)
        close (13)
        close (14)
        close (15)
        close (16)

        do r=1,ndrg
        if(iflr(r).eq.0) then
        n=dmnpnt(r)
        write(*,222) nndnme(n)
        call errmsg(4,508)
        endif
        enddo
```

Step 4a: Read in industrial demand data. Open 'ind_ld.spc' and 'ind_dem.spc'. 'ind_dem.spc' contains industrial demand values. 'ind_ld.spc' contains industrial demand load factors.

```
        call read_hdr(11,'ind_ld.spc',3)
        open(24,file='ind_dem.spc')

        do 410 r=1,ndrg
        do 405 ss=1,mxniss
        iflrs(r,ss)=0
405     continue
410     continue
        niss=0
```

Step 4b: Read in 5-year data from 'ind_dem.spc'.

```
420     read(24,421,end=460) nname,sname,bsedem,efff,
*       (ttinpq(t),t=1,mxnval),(ttinpe(t),t=1,mxnval),
*       (linp(l),l=1,mxnlds),ishr,
        ((ttshr(t,ii),t=1,mxnval),ii=1,6)
421     format(a18,2x,a20,f7.1,1x,f6.2,1x,7(f7.1,1x),7(f6.2,1x),
*       7(f6.3,1x),f6.2/21(f6.2,1x)/21(f6.2,1x))
```

Step 4c: Calculate growth rates to interpolate annual demand values.

```
        do 423 i=1,5
        tqrate(i)=(((ttinpq(i+2)/ttinpq(i+1))**(0.2))-1.0)*100.0
        terate(i)=(((ttinpe(i+2)/ttinpe(i+1))**(0.2))-1.0)*100.0
        do 422 ii=1,6
        tsrate(i,ii)=
*       (((ttshr(i+2,ii)/ttshr(i+1,ii))**(0.2))-1.0)*100.0
422     continue
423     continue

        do 425 t=1,mxntme
        call growth(ttinpq(2),tqrate,t,tiqval)
        tinpq(t)=tiqval
        call growth(ttinpe(2),terate,t,tieval)
        tinpe(t)=tieval
        do 424 ii=1,6
        do j=1,5
        tsrate2(j)=tsrate(j,ii)
        enddo
        call growth(ttshr(2,ii),tsrate2,t,tisval)
        tshr(t,ii)=tisval
424     continue
425     continue
```

Step 4d: Normalize industrial product shares.

```
        do t=1,ntme
        vscl=(tshr(t,2)+tshr(t,3)+tshr(t,4)+tshr(t,5)+tshr(t,6))/100.0
        if((vscl.ge.0.85).and.(vscl.le.1.15)) then
        do ii=2,6
        tshr(t,ii)=tshr(t,ii)/vscl
        enddo
        endif
        enddo

        bsedem=tinpq(1)
```

Note: Load factors linp(l) are read from ‘ind_ld.spc’ and overwrite those from ‘ind_dem.spc.’

Step 4e: Read in industrial load factors from ‘ind_ld.spc’.

```
        read(11,*) nname2,(linp(l),l=1,nlds)
        do l=nlds+1,mxnlds
        linp(l) = 0.0
        enddo
        if (nname.ne.nname2) then
        write(*,*) 'region names/region order are inconsistent between'
        write(*,*) 'ind_dem.spc and ind_ld.spc'
        write(*,*) 'program aborted'
        stop
        endif
        write(99,*)'Industrial Loads'
        write(99,*)nname2,(linp(l),l=1,mxnlds)

        do 430 r=1,ndrg
        n=dmnpnt(r)
        if(nndnme(n).eq.nname) go to 431
        430 continue
        write(*,222) nname
        call errmsg(4,513)
```

```
431     ss=0
432     ss=ss+1
        if(ss.gt.niss) go to 434
        if(idmssn(ss).eq.sname) go to 435
        go to 432
434     niss=niss+1
        if(niss.gt.mxniss) call errmsg(4,514)
        idmssn(ss)=sname
```

Step 4f: Adjust industrial data for economic growth.

```
435     if(iflrs(r,ss).ne.0) then
        write(*,222) nname
        call errmsg(4,515)
        endif
        iflrs(r,ss)=1
        idmbqn(r,ss)=bsedem
        idmeff(r,ss)=efff
        idmish(r,ss)=ishr
        do 440 t=1,ntme
        vscl=(econsp(t,r)/tinpe(t))**efff
        idmefc(t,r,ss)=vscl*tinpq(t)
440     continue
        vscl=0.0
        do 445 l=1,nlds
        linp(l)=amax1(linp(l),0.0)
        vscl=vscl+linp(l)*ldsdy(l)/365.0
445     continue
        do 450 l=1,nlds
        if(vscl.le.0.0) then
        idmldf(l,r,ss)=1.0
        else
        idmldf(l,r,ss)=linp(l)/vscl
        endif
450     continue

        do 455 t=1,ntme
        idmshr(t,r,ss,1)=amax1(0.0,amin1(tshr(t,1),100.0))/100.0
        vscl=0.0
        do 452 ii=2,6
        tshr(t,ii)=amax1(0.0,amin1(tshr(t,ii),100.0))/100.0
        vscl=vscl+tshr(t,ii)
452     continue
        if((vscl.le.0.999).or.(vscl.ge.1.001)) then
        write(*,453) nname,sname,tme(t),vscl
453     format(' Region: ',a20,' Ind-Subsector: ',a20,' Year: ',i4,
*       ' Value: 'f8.6)
        call errmsg(3,516)
        endif
        do 454 ii=2,6
        if(vscl.le.0.0) then
        idmshr(t,r,ss,ii)=0.2
        else
        idmshr(t,r,ss,ii)=tshr(t,ii)/vscl
        endif
454     continue
455     continue
        go to 420
```

Step 4g: Close ‘ind_dem.spc’ and ‘ind_ld.spc’.

```
460     close(24)
        close(11)

        do 470 r=1,ndrg
        do 465 ss=1,niss
        if(iflrs(r,ss).eq.0) then
        n=dmnpnt(r)
        write(*,453) nndnme(n),idmssn(ss)
        call errmsg(3,517)
        endif
465     continue
470     continue
```

Note: mxnval is the number of values read in from non-annualized files.
mxnlds is the maxinum number of load segments.
mxntme is the maxinum number of time periods.
elecsp is the specification of future growth.
euexc is the existing generation capacity by fuel type.
euovcf is the load factor for existing capacity.
eushr is the share of demand in four load segments.
eutot is the total demand for electricity in the region.

Step 5a: Read in electric utility demand. Open ‘eu1_ld.spc’, ‘eu2_ld.spc’, ‘eu3_ld.spc’, ‘eu4_ld.spc’, ‘eu_dem.spc’. ‘eu_dem.spc’ contains electricity demand figures. ‘eu1_ld.spc’, ‘eu2_ld.spc’, ‘eu3_ld.spc’, ‘eu4_ld.spc’ contain electric utility load factors in the following manner:

‘eu1_ld.spc'... electric utility season 1, gas seasons 1,...,nlds
‘eu2_ld.spc'... electric utility season 2, gas seasons 1,...,nlds
‘eu3_ld.spc'... electric utility season 3, gas seasons 1,...,nlds
‘eu4_ld.spc'... electric utility season 4, gas seasons 1,...,nlds

```
        open(11,file='eu1_ld.spc')
        open(12,file='eu2_ld.spc')
        open(13,file='eu3_ld.spc')
        open(14,file='eu4_ld.spc')

        open(25,file='eu_dem.spc')
        do 510 r=1,ndrg
        iflr(r)=0
        do 505 f=1,7
        iflreu(r,f)=0
505     continue
510     continue

520     read(25,521,end=560) nname,dflg,fname,(eshr(l),l=1,4),
*       (ttinp(t),t=1,mxnval),((linpe(l1,l),l1=1,mxnlds),l=1,4),
*       (ttinpe(t),t=1,mxnval)
521     format(a20,2a8,4(f5.2,1x),7(f6.2,1x),28(f4.2,1x),7(f5.3,1x))
        vscl=(eshr(1)+eshr(2)+eshr(3)+eshr(4))/100.0
```

Step 5b: Calculate growth rates to interpolate annual values.

```
do i=1,5
if((ttinp(i+2).eq.0.0).and.(ttinp(i+1).eq.0.0)) then
tcapg(i)=0.0
else
tcapg(i)=(((ttinp(i+2)/ttinp(i+1))**(0.2))-1.0)*100.0
endif
if((ttinpe(i+2).eq.0.0).and.(ttinpe(i+1).eq.0.0)) then
tmutg(i)=0.0
else
tmutg(i)=(((ttinpe(i+2)/ttinpe(i+1))**(0.2))-1.0)*100.0
endif
enddo

do t=1,mxntme
call growth(ttinp(2),tcapg,t,tcapval)
tinp(t)=tcapval
call growth(ttinpe(2),tmutg,t,tmutval)
tinpe(t)=tmutval
enddo
```

Step 5c: Read in load factors linp(l) from ‘eu1_ld.spc’, ‘eu2_ld.spc’, ‘eu3_ld.spc’, ‘eu4_ld.spc’. These factors overwrite those from ‘eu_dem.spc’.

```
if (dflg.ne.'TOT-ELEC') go to 522 ! only need to match one set of node names

read(11,*) nname2,(linpe(l1,1),l1=1,nlds)
do l=nlds+1,mxnlds
linpe(l,1) = 0.0
enddo
if (nname.ne.nname2) then
write(99,*) 'region names/region order are inconsistent between'
write(99,*) 'eu_dem.spc and eu1_ld.spc'
write(99,*) 'program aborted'
stop
endif

read(12,*) nname2,(linpe(l1,2),l1=1,nlds)
do l=nlds+1,mxnlds
linpe(l,2) = 0.0
enddo

if (nname.ne.nname2) then
write(99,*) 'region names/region order are inconsistent between'
write(99,*) 'eu_dem.spc and eu2_ld.spc'
write(99,*) 'program aborted'
stop
endif

read(13,*) nname2,(linpe(l1,3),l1=1,nlds)
do l=nlds+1,mxnlds
linpe(l,3) = 0.0
enddo

if (nname.ne.nname2) then
write(99,*) 'region names/region order are inconsistent between'
write(99,*) 'eu_dem.spc and eu3_ld.spc'
```

```
        write(99,*) 'program aborted'
        stop
        endif

        read(14,*) nname2,(linpe(l1,4),l1=1,nlds)
        do l=nlds+1,mxnlds
        linpe(l,4) = 0.0
        enddo

        if (nname.ne.nname2) then
        write(99,*) 'region names/region order are inconsistent between'
        write(99,*) 'eu_dem.spc and eu4_ld.spc'
        write(99,*) 'program aborted'
        stop
        endif

522     continue
        if((vscl.lt.0.99).or.(vscl.gt.1.01)) then
        write(*,*)'vscl=',vscl
        write(*,524) nname,dflg,fname
524     format(' Region: ',a20,' Type: ',a8,' Fuel: ',a8)
        call errmsg(4,518)
        endif
```

Step 5d: Assign fuel index to appropriate fuel type.

```
        if(fname.eq.'NUCLEAR ') then
        f=1
        elseif(fname.eq.'COAL    ') then
        f=2
        elseif(fname.eq.'HYDRO/OT') then
        f=3
        elseif(fname.eq.'COMB CYL') then
        f=4
        elseif(fname.eq.'O/G LS-R') then
        f=5
        elseif(fname.eq.'O/G HS-R') then
        f=6
        elseif(fname.eq.'O/G DIST') then
        f=7
        else
        f=0
        endif
```

Step 5e: Find the pointer to the current node name.

```
        do 530 r=1,ndrg
        n=dmnpnt(r)
        if(nndnme(n).eq.nname) go to 535
530     continue
        write(*,222) nname
        call errmsg(4,519)

535     if(dflg.eq.'TOT-ELEC') then
        if(iflr(r).ne.0) then
        write(*,222) nname
        call errmsg(4,520)
        endif
        iflr(r)=1
```

Step 5f: Assign total electricity demand values.

```
        do 540 t=1,ntme
        eutot(r,t)=tinp(t)
540     continue
```

Step 5g: Assign shares for electrical power load segments.

```
        do 545 l=1,4
        eushr(r,l)=eshr(l)
        do 543 l1=1,nlds
        euldf(l1,r,l)=linpe(l1,l)
543     continue
545     continue
        elseif(dflg.eq.'EX-CAP  ') then

        if((f.eq.0).or.(iflreu(r,f).ne.0)) then
        write(*,524) nname,dflg,fname
        call errmsg(4,521)
        endif
        iflreu(r,f)=1
```

Step 5h: Calculate a maximum electricity generation value for each region, fuel type, and year.

```
        do 550 t=1,ntme
        euexc(r,f,t)=tinp(t)*(8760.0*tinpe(t))/1000.0
        euovcf(r,f,t)=tinpe(t)
550     continue
```

Step 5i: Store share of electricity capacity by region, fuel type, and electricity season.

```
        do 555 l=1,4
        eueshr(r,f,l)=eshr(l)
555     continue
        else
        write(*,556) nname,dflg
556     format(' Region: ',a20,' Code: ',a8)
        call errmsg(4,524)
        endif
        go to 520
```

Step 5j: Close electricity utility files.

```
560     close(25)

        close(11)
        close(12)
        close(13)
        close(14)
```

Step 5k: Make sure each region was read in.

```
        do 570 r=1,ndrg
        if(iflr(r).eq.0) then
```

```
      n=dmnpnt(r)
      write(*,222) nndnme(n)
      call errmsg(4,523)
      endif
      do 565 f=1,7
      if(iflreu(r,f).eq.0) then
      n=dmnpnt(r)
      write(*,561) nndnme(n),f
561   format(' Region: ',a20,' Fuel Index: ',i3)
      call errmsg(4,524)
      endif
565   continue
570   continue
```

Step 5l: **Normalize the regional electricity demand to match the total electricity demand.**

```
      do 580 t=1,ntme
      tinp(t)=0.0
      do 575 r=1,ndrg
      if(eutot(r,t).le.0.0) call errmsg(4,525)
      tinp(t)=tinp(t)+eutot(r,t)
575   continue
      if(tinp(t).le.0.0) call errmsg(4,525)
580   continue
      do 590 r=1,ndrg
      do 585 t=ntme,1,-1
      eutot(r,t)=eutot(r,t)*elecsp(t)/tinp(t)
585   continue
590   continue
```

Note:

vcap1,2 is the capital cost of the powerplant.
vfom1,2 is the fixed annual O&M for the powerplant.
vvom1,2 is the variable O&M for the powerplant.
vhr1,2 is the heat rate for the powerplant.
vccr is the capital charge rate.
eunfcs is the non-fuel generation costs by fuel type load segment.
eunvcs is the non-fuel variable costs.
euflef is the generation efficienc y by fuel type, load segment
eunvcf is the load factor specification for new capacity.
eusox is the sox pollution add on cost.
euoenv is the other environmental cost add on.

Step 6a: **Open file and read in variables. ‘eu_gen.spc’ contains the electric utility cost and efficiency data by plant type.**

```
      open(26,file='eu_gen.spc')
      do 605 f=1,7
      iflreu(1,f)=0
605   continue

620   read(26,621,end=660) fname,vcap1,vfom1,vvom1,vhr1,vcap2,vfom2,
*     vvom2,vhr2,vccr,(vcf(l),l=1,4),tsox,toenv
621   format(a8,2(f6.1,1x,f5.2,1x,f5.2,1x,f7.1,1x),f5.2,1x,4(f5.2,1x),
*     2f7.4)
```

Step 6b: Assign fuel index.

```
        if(fname.eq.'NUCLEAR ') then
        f=1
        elseif(fname.eq.'COAL    ') then
        f=2
        elseif(fname.eq.'HYDRO/OT') then
        f=3
        elseif(fname.eq.'COMB CYL') then
        f=4
        elseif(fname.eq.'O/G LS-R') then
        f=5
        elseif(fname.eq.'O/G HS-R') then
        f=6
        elseif(fname.eq.'O/G DIST') then
        f=7
        else
        write(*,622) fname
622     format(' Fuel Type: ',a8)
        call errmsg(4,525)
        endif
```

Step 6c: Set read-in flag = 1 (fuel type data read).

```
        if(iflreu(1,f).ne.0) then
        write(*,622) fname
        call errmsg(5,526)
        endif
        iflreu(1,f)=1
```

Step 6d: Calculate fixed O&M, variable O&M, capital costs, and heat rates for existing and new plants by fuel type.

```
        do 630 l=1,4
        val=1.0/(8760.0*vcf(l))
        eunfcs(f,l,1)=vcap1*vccr/100.0*val+vfom1*val
        eunvcs(f,l,1)=vvom1/1000.0
        euflef(f,l,1)=(vhr1/1000.0)/1.032
        eunfcs(f,l,2)=vcap2*vccr/100.0*val+vfom2*val
        eunvcs(f,l,2)=vvom2/1000.0
        euflef(f,l,2)=(vhr2/1000.0)/1.032
        eunvcf(f,l)=vcf(l)
        if(vcf(l).ne.vcf(1)) call errmsg(4,528)
630     continue
        eusox(f)=tsox
        euoenv(f)=toenv
        go to 620
```

Step 6e: Make sure each fuel type was read in.

```
660     close(26)
        do 670 f=1,7
        if(iflreu(1,f).ne.1) then
        write(*,661) f
661     format(' fuel code: ',i2)
        call errmsg(4,527)
        endif
670     continue
        return
        end
```

SUBROUTINES FOUND IN INTRDD

SUBROUTINE GROWTH (BASE,RATE0,YR,VALUE)

CALLED BY: INTRDD.FOR (reads in demand specifications from the demand input files and writes out the appropriate demand data to the LP files)

CALLS: None

READS: None

CREATES: None

MAIN THEME: Growth calculates the rates over time with growth factors read in from a file.

Step 1a: **Calculate growth rates by year.**

```
        Implicit None
        Include 'intgm.cmn'
        real base,rate0(5),rates(5),value,rateone
        integer yr,ii

        do ii=1,5
        rates(ii) = 1.0+rate0(ii)/100.0
        enddo
```

Step 1b: **Apply growth rates to calculate annual forecasts.**

```
        [1993,1994]
        if ((tme(yr).ge.1993).and.(tme(yr).le.1994)) then
        if (abs(rates(1)).le.1.d-6) then
        value = base
        else
        value = base*rates(1)**(tme(yr)-1995)
        endif


        return

        [1995,2000]
        else if ((tme(yr).ge.1995).and.(tme(yr).le.2000)) then
        value = base*rates(1)**(tme(yr)-1995)
        return


        (2000,2005]
        else if (tme(yr).le.2005) then
        value = (base*rates(1)**5)*
*       (rates(2)**(tme(yr)-2000))
```

```
      return

      (2005,2010]
      else if (tme(yr).le.2010) then
      value = (base*rates(1)**5)*
1     (rates(2)**5)*
2     (rates(3)**(tme(yr)-2005))

      return

      (2010,2015]
      else if (tme(yr).le.2015) then
      value = (base*rates(1)**5)*
1     (rates(2)**5)*
2     (rates(3)**5)*
3     (rates(4)**(tme(yr)-2010))
      return


      else ! (tme(yr).gt.2015)
      value = (base*rates(1)**5)*
1     (rates(2)**5)*
2     (rates(3)**5)*
3     (rates(4)**5)*
3     (rates(5)**(tme(yr)-2015))

      return
      end if

      end
```

SUBROUTINE READ_HDR (FILENUM,FILENAME,LINES)

CALLED BY: GASHIST.FOR (contains the gas historical price routine and reads in gas prices, by track, from 1993 to the beginning year of the GSAM Integrating Model)
INTRDD.FOR (reads in demand specifications from the demand input files and writes out the appropriate demand data to the LP files)
INTRDT.FOR (reads in the transportation and other non-demand data for the LP)

CALLS: None

READS: None

CREATES: None

MAIN THEME: This subroutine opens a file and reads the appropriate number of header lines.

Step 1: **Open file and read the appropriate number of header lines.**

```
Implicit None
integer filenum,lines,i
character*11 filename
character*80 dummy

open(unit=filenum,file=filename)
do i=1,lines
read(filenum,*)dummy
enddo
return
end
```

SUBROUTINE: INTRDS.FOR

CALLED BY: INTRDT.FOR (reads in transportation and other non-demand data for the LP)

CALLS: ERRMSG.FOR (prints out error and warning messages)

READS: **‘**supply.spc’ (contains supply price and quantity data)

CREATES: None

MAIN THEME: This subroutine reads in supply increment specifications.

Note: isps is the pass number of supply increment
ndd is used to count days in estimating supply load factors
l1 is the index to load period
nname is used to read node names
tpprc(t),tptot(t) is used to read in supply increment data.
valmn,valmx,valu,vald is used to calculate load factors.
mxnsin is the maxinum supply increment.
mxntme is the maxinum number of time periods.
nsup is the number of supply increments.
nsps is the index of pass through the model of new supply.
suppas is the pass of supply increment.
mxnsrg is the maxinum number of supply regions.
supprc is the annual average price of supply by region and year.
suptot is the supply for the load segment at the specified price.
supldf is the load factors for supply.
ldsdy is the number of days in each load segment.

Step 1a: **Set supply increments to zero and open ‘supply.spc’. ‘supply.spc’ contains supply price and quantity data.**

```
      nsup=0
      nsps=1
      open(16,file='supply.spc')
      do 50 k=1,mxnsin
      suppas(k)=0
      do 40 s=1,mxnsrg
      do 30 t=1,mxntme
      supprc(t,s,k)=0.
      suptot(t,s,k)=0.
30    continue
40    continue
50    continue
      koff=0
```

Step 1b: Read in data from ‘supply.spc’ and fill in supply arrays.

```
        read(16,*) dummy
100     read(16,101,end=190) nname,k,(tpprc(t),tptot(t),t=1,ntme)
101     format(a20,i3,33(f6.2,f7.1))

        isps=1
        if((k.lt.nsup).or.(k.gt.(nsup+1))) call errmsg(4,201)
```

Step 1c: Pack out early supply vectors if overflow exists.

```
        if(((k+koff).gt.mxnsin).or.
        ((nsup.gt.0).and.(isps.ne.suppas(1)))) then
        call errmsg(2,201)
        ispsx=suppas(1)
        k2=0
        do 110 k1=1,mxnsin
        if((suppas(k1).ne.ispsx).and.(k1.le.(k+koff))) then
        k2=k2+1
        suppas(k2)=suppas(k1)
        do 105 t=1,ntme
        do 103 s=1,nsrg
        supprc(t,s,k2)=supprc(t,s,k1)
        suptot(t,s,k2)=suptot(t,s,k1)
103     continue
105     continue
        endif
110     continue
        koff=k2-k
        endif
        if((isps.lt.nsps).or.(isps.gt.(nsps+1))) call errmsg(4,202)
        nsup=max0(nsup,k)
        nsps=max0(nsps,isps)
        k=k+koff
        if(suppas(k).eq.0) then
        suppas(k)=isps
        else
        if(suppas(k).ne.isps) call errmsg(4,204)
        endif
        do 120 s=1,nsrg
        n=suppnt(s)
        if(nname.eq.nndnme(n)) go to 130
120     continue
        call errmsg(4,205)
130     do 150 t=1,mxntme
        supprc(t,s,k)=tpprc(t)
        suptot(t,s,k)=tptot(t)*1000.0/365.0
150     continue
        go to 100

190     nsup=nsup+koff
```

Step 1d: Set up load factor arrays.

```
        do 250 s=1,nsrg
        valmx=suplfc(s)
        valmn=1.0/suplfc(s)
        do 210 l=1,nlds
        supldf(s,l)=1.0
```

```
210     continue
        if(nlds.gt.1) then
        ndd=0
        do 230 l=1,nlds-1
        valu=(valmx-supldf(s,l))*float(ldsdy(l))
        vald=valu/float(365-ndd-ldsdy(l))
        do 215 l1=l+1,nlds
        vald=amin1(vald,(supldf(s,l1)-valmn))
215     continue
        vald=amax1(vald,0.0)
        valu=vald*float(365-ndd-ldsdy(l))/float(ldsdy(l))
        supldf(s,l)=supldf(s,l)+
        do 220 l1=l+1,nlds
        supldf(s,l1)=supldf(s,l1)-vald
220     continue
        ndd=ndd+ldsdy(l)
230     continue
        endif
250     continue
        call read_slf
```

Step 2a: Determine "Average" prices for each supply vector and create continuous list of vectors.

```
        k1=nsup
        k2=(nsup-1)*9+nsup
        if(k2.gt.mxnsin) call errmsg(4,201)
300     suppas(k2)=suppas(k1)
        do 350 s=1,nsrg
        do 340 t=1,ntme
        supprc(t,s,k2)=supprc(t,s,k1)
        suptot(t,s,k2)=suptot(t,s,k1)
        val=1.0
        do 330 k3=1,9
        k4=k2-k3
        val=val-0.1
        supprc(t,s,k4)=val*supprc(t,s,k2)+(1.0-val)*supprc(t,s,k1-1)
        suptot(t,s,k4)=val*suptot(t,s,k2)+(1.0-val)*suptot(t,s,k1-1)
        suppas(k4)=suppas(k2)
330     continue
340     continue
350     continue
        k2=k2-10
        k1=k1-1
        if(k1.gt.1) go to 300
        nsup=(nsup-1)*9+nsup

        do 400 s=1,nsrg
        do 390 t=1,ntme
        if(supprc(t,s,1).le.supprc(t,s,nsup)) then
        ko=1
        kd=1
        ks=2
        ke=nsup
        else
        ko=nsup
        kd=-1
        ks=nsup-1
        ke=1
        endif
        supavp(t,s,ko)=supprc(t,s,ko)
        if(nsup.gt.1) then
        do 380 k=ks,ke,kd
        if(suptot(t,s,k).gt.0.0) then
        supavp(t,s,k)=(suptot(t,s,ko)*supavp(t,s,ko)+
*       (suptot(t,s,k)-suptot(t,s,ko))*supprc(t,s,k))/
*       suptot(t,s,k)
        lse
         supavp(t,s,k)=0.0
        endif
        ko=k
380     continue
        endif
390     continue
400     continue
```

Step 2b: Close ‘supply.spc’.

```
        close(16)
        return
        end
```

SUBROUTINE: INTRDT.FOR

CALLED BY:	INTMGN.FOR (translates LP demand, pipeline, storage and other data into MPS format to be used as input to the LP solver) INTRPT.FOR (produces output files) INTRVS.FOR (reads the solution and determines if additional supply vectors are required and if so generates addition supply vectors)
CALLS:	ERRMSG.FOR (prints out error and warning messages) INTRDD.FOR (reads in demand specifications from the demand input files and writes out the appropriate demand data to the LP files) INTRDS.FOR (reads in supply increment specifications) PEAK_SUP (reads in peak supply data) PPP.FOR (calculates the petroleum product prices) READ_HDR (reads the appropriate number of header line in a file) SROM.FOR (read in storage reservoir-level information and determines which of the three storage options is most economically advantageous to the storage reservoir operator) ZAP.FOR (zeros out an array)
READS:	**‘**coal_pr.spc’ (contains the base coal price in 1995 and the forecasted growth rates through the year 2020) ‘crude.spc’ (contains the historical and forecasted crude prices from 1993 to 2025) ‘dual_prc.spc’ (contains the price of natural gas according to season and region) ‘dmn_sec.spc’ (contains the mark-up factors by region for each sector) ‘gen_tml.spc’ (contains the time specifications, discount rate, model start year, end year, and current year, and the number of days in the seasons) ‘link_nde.spc’ (contains capacity, costs, fuel requirements, and expansion factors for pipelines) ‘node.spc’ (defines the supply, demand, and other region names and indicators) ‘oth_sup.spc’ (contains supply specifications for LNG, ANGST, and other supply projects with unique economic factors and constraints)
CREATES:	None
MAIN THEME:	This subroutine reads in the transportation and other non-demand data for the LP.

Note: ndy is the count on the number of days.
nname,nnamo,nnamd is used to read node names.
sname is used to read sector and project names.
val is used to calculate fraction.
tplfc is used to read supply load factor.
begyr,endyr is the beginning year and end year for model run.
machep is our version of machine epsilon (smallest positive number) used to avoid problems in the LP.

```
          machep = 0.001
```

Step 1a: **Open file and read in variables. begyr, endyr are the actual beginning and ending years (i.e., 1997, 2010). 'gen_tml.spc' contains the time period specifications, discount rate, model start year, end year, and current year, and the number of days in the seasons.**

Note: ibegyr, iendyr, are the indices to these years starting with 1993 =1 thus if begyr =1997 => ibegyr = 5 (1997-1993+1) and if endyr =2010 => ibegyr = 8 (2010-1993+1) and if curyr =1998 => icuryr = 6 (1998-1993+1).

```
          open(11,file='gen_tml.spc')
          read(11,101) begyr,endyr,curyr
          read(11,102) (ldsdy(l),l=1,mxnlds)
          read(11,104) disrte
101       format(3(i4,1x))
102       format(7(i3,1x))
104       format(f4.1)
```

Step 1b: **Close file.**

```
          close(11)
```

Step 1c: **Calculate ibegyr, iendyr. These are the indices of the beginning and end years, respectively.**

Note: ibegyr = 1 corresponds with 1993, ibegyr = 2 corresponds with 1994,…

```
          ntme=mxntme
          ibegyr =begyr-1993+1
          iendyr =endyr-1993+1
          icuryr =curyr-1993+1

          if(ntme.gt.mxntme) call errmsg(4,101)
          if(begyr.gt.endyr) call errmsg(4,101)

          do 120 t=1,ntme
```

```
        tme(t)=1993-1+t
120     continue
```

Step 1d: Check for valid values.

```
        nlds=mxnlds
        ndy=0
        do 130 l=1,mxnlds
        if(ldsdy(l).ne.0) then
        if((l.gt.1).and.(ldsdy(l-1).eq.0)) call errmsg(4,102)
        ndy=ndy+ldsdy(l)
        else
        nlds=min0(nlds,l-1)
        endif
130     continue
        if(nlds.le.0) call errmsg(4,102)
        if(ndy.ne.365) call errmsg(4,102)
```

Note: prccrd is the price of crude.

Step 2a: Open file and read in the appropriate crude prices and year. 'crude.spc' contains the historical and forecasted crude prices from 1993 to 2025. Close file.

```
        open(11,file='crude.spc')
        read(11,*)
        do 145 t=1,ntme
140     read(11,141) tyear,prccrd(t)
141     format(i4,1x,f5.2)
145     continue

        close(11)
```

Step 3a: Open file. 'node.spc' defines the supply, demand, and other region names and indicators.

```
        open(12,file='node.spc')
        nnde=0
        nsrg=0
        ndrg=0

        do 210 s=1,mxnsrg
        suppnt(s)=0
        suplfc(s)=0.0
        gthcst(s)=0.0
210     continue

        do 215 r=1,mxndrg
        dmnpnt(r)=0
215     continue

220     read(12,221,end=230) nname,s,r,tplfc,val
221     format(a20,2(i2,1x),f6.3,1x,f6.3)
        nnde=nnde+1

        write(*,*)nname,nnde
```

```
          if(nnde.gt.mxnnde) call errmsg(4,103)
          nndnme(nnde)=nname
```

Step 3b: Check and assign supply node information.

Note: nsrg is the number of GSAM supply regions as specified in 'node.spc'.

```
          if(s.gt.0) then
          if(s.gt.mxnsrg) call errmsg(4,104)
          if(suppnt(s).ne.0) call errmsg(4,105)
          if(tplfc.lt.1.0) call errmsg(4,203)
          suppnt(s)=nnde
          suplfc(s)=tplfc
          gthcst(s)=val
          nsrg=max0(nsrg,s)
          endif
```

Step 3c: Check and assign demand node information.

Note: ndrg is the number of GSAM demand regions as specified in 'node.spc'.

```
          if(r.gt.0) then
          if(r.gt.mxndrg) call errmsg(4,106)
          if(dmnpnt(r).ne.0) call errmsg(4,107)
          dmnpnt(r)=nnde
          ndrg=max0(ndrg,r)
          endif

          go to 220
```

Step 3d: Close file and and check for valid values.

```
230       close(12)
          if(nsrg.eq.0) call errmsg(4,108)
          if(ndrg.eq.0) call errmsg(4,109)
          do 235 s=1,nsrg
          if(suppnt(s).eq.0) call errmsg(4,110)
235       continue
          do 240 r=1,ndrg
          if(dmnpnt(r).eq.0) call errmsg(4,111)
240       continue
```

Step 4a: Call PPP.FOR. PPP.FOR calculates the appropriate petroleum product prices by region.

```
          call ppp
```

Step 5a: **Open file ‘coal.spc’ and initialize coal prices to zero. ‘coal_pr.spc’ contains the base coal price in 1995 and the forecasted growth rates through the year 2020.**

```
        call read_hdr(11,'coal_pr.spc',5)
        do 250 r=1,ndrg
        do 245 t=1,ntme
        coalpr(t,r)=0.0
245     continue
250     continue
```

Step 5b: **Read in the growth rates and calculate coal prices for each year. Check for valid values and close file.**

```
260     read(11,*,end=280) nname,tcbse,(tcrates(t),t=1,5)
        do 270 r=1,ndrg
        n=dmnpnt(r)
        if(nname.ne.nndnme(n)) go to 270
        do 265 t=1,ntme
        call growth(tcbse,tcrates,t,tcprc)
        coalpr(t,r)=tcprc
265     continue
        go to 260

270     continue
        write(*,271) nname
271     format(' Demand region: ',a20)
        call errmsg(3,204)
        go to 260
280     close(11)
```

Note:

nlnk is the number of transport links.
mxnlnk is the maxinum number of links.
nnamo, nname, nnamd are used to read node names.
lnkcap is the capacity of the link.
lnkcss is the levelized investment costs.
lnkfom is the annual fixed O&M costs for the link.
lnkfyr is the first year that link capacity is available.
mxntad is the maxinum number of transport additions allowed per link.
lnkvom is the annual variable O&M costs for the link.
lnkfus is the fuel use percentage for the transport capacity increments.
lnkpnt is the pointer to the origin and destination nodes for the link.

Step 6a: **Open file and read in data. ‘link_nde.spc’ contains capacity, costs, and fuel requirements for existing as well as potential pipelines.**

```
        open(13,file='link_nde.spc')
        nlnk=0
300     nlnk=nlnk+1
        if(nlnk.gt.mxnlnk) call errmsg(4,112)
        read(13,301,end=350) nnamo,nnamd,(lnkcap(m,nlnk),lnkccs(m,nlnk),
*       lnkfom(m,nlnk),lnkfyr(m,nlnk),m=1,mxntad),
*       lnkvom(nlnk),lnkfus(nlnk),cnus(nlnk)
301     format(2a20,2(f7.1,1x,f6.1,1x,f6.1,1x,i4,1x),f6.3,1x,f5.2,f5.2)
```

Step 6b: **Reset zero variable O&M values to machine epsilon to avoid problems in the LP.**

```
        if (lnkvom(nlnk).le.0.0) lnkvom(nlnk) = machep
```

Step 6c: **Check for valid values.**

Note: curyr-1 is the last year of history matching .
History matching is only for existing capacity.

```
        if(lnkfyr(1,nlnk).lt.curyr) lnkfyr(1,nlnk)=0
        lnkpnt(1,nlnk)=0
        lnkpnt(2,nlnk)=0

        do 310 n=1,nnde
        if(nnamo.eq.nndnme(n)) lnkpnt(1,nlnk)=n
        if(nnamd.eq.nndnme(n)) lnkpnt(2,nlnk)=n
310     continue
```

Step 6d: **Check for valid pipeline capacity additions.**

```
        if((lnkpnt(1,nlnk).eq.0).or.(lnkpnt(2,nlnk).eq.0)) then
        write(*,311) nnamo,nnamd
311     format(' link - origin: ',a20,' destination: ',a20)
        call errmsg(4,113)
        endif
        do 320 m=2,mxntad
        if(lnkcap(m,nlnk).ne.0.0) then
        if(lnkfyr(m-1,nlnk).gt.lnkfyr(m,nlnk)) then
        write(*,311) nnamo,nnamd
        call errmsg(4,114)
        endif
        endif
320     continue

        go to 300
```

Step 6e: **Close file. Check for valid values.**

```
350     close(13)
        nlnk=nlnk-1
        if(nlnk.le.0) call errmsg(4,115)
```

Note: ‘dual_prc.spc’ is the file containing the (undiscounted) dual prices from the material balance constraints.

Step 7a: **Open ‘dual_prc.spc’ and read in the dual prices. ‘dual_prc.spc’ contains the price of natural gas according to season, time, and region. Close ‘dual_prc.spc’.**

```
        open(unit=29,file='dual_prc.spc')
380     read(29,*,end=400) n,t,l,duals(n,t,l)
        go to 380
400     close(29)
```

Step 8a: **Call SROM.FOR. SROM.FOR is the routine for the Storage Reservoir Operator Module. This subroutine reads in storage reservoir-level information and determines which of the three storage options is most profitable to the storage reservoir operator.**

```
        call srom
```

Note: dmsnme is the name of the demand sector.
dmsdmr is the distribution margin (end-use minus wholesale) by region.
mxndms is the maxinum number of demand sectors.
mxndrg is the maxinum number of demand regions.
ndms is the number of demand sectors.
dmnpnt is the pointer to the node where the demand region is located.

Step 9a: **Open ‘dmn_sec.spc’. ‘dmn_sec.spc’ contains the mark-up factors for each sector by region.**

```
        Open(14,file='dmn_sec.spc')
        ndms=4
        dmsnme(1)='Residential        '
        dmsnme(2)='Commercial'
        dmsnme(3)='Industrial'
        dmsnme(4)='Elec-Gen  '
        call zap(dmsdmr,mxndms,mxndrg,1,1,1)
460     read(14,461,end=480) nname,sname,tcst
461     format(2a20,1x,f6.2)
        c=0
        if(ndms.gt.0) then
        do 465 c=1,ndms
        if(dmsnme(c).eq.sname) go to 470
465     continue
        c=0
470     continue
        endif
```

Step 9b: Find the current sector of ‘dmn_sec.spc’.

```
        if(c.eq.0) then
        ndms=ndms+1
        if(ndms.gt.mxndms) call errmsg(4,120)
        dmsnme(ndms)=sname
        c=ndms
        endif
```

Step 9c: Find the pointer to the current node name.

```
        do 475 r=1,ndrg
        n=dmnpnt(r)
        if(nname.ne.nndnme(n)) go to 475
        dmsdmr(c,r)=tcst
        go to 460
475     continue
        call errmsg(4,120)
```

Step 9d: Close ‘dmn_sec.spc’.

```
480     close(14)
        if(ndms.le.0) call errmsg(4,120)
```

Step 10a: Call INTRDD.FOR. This routine reads in demand specifications from the demand input files and creates arrays/variables neccessary for implementing demand within the Integrating LP.

```
        call intrdd
```

Step 11a: Call INTRDS.FOR. INTRDS.FOR is a subroutine that reads in supply increment specifications.

```
600     call intrds
```

Note: npks is the number of peak shaving options.

indices:
p is the peak supply source (p=1 propane, p=2 LNG).
n is the node number.
t is the time period.
l is the gas load seasons.
k is the status (k=1 existing, k=2 new).

variables:
PKOpntlk is the operating level.
PKIpntk is the investment level.

constants:
pkvc(p,n,k) is the variable cost.
pklc(p,n,k) is the levelized investment cost.
pkfc(p,n,k) is the fixed O&M costs.
ldsdy(l) is the number of days in gas load seasons l.
pkd(p,n,k) is the maximum deliverability.
pkfyr(p,n,k) is the first year available.
pksc(p,n,k) is the storage capacity (maximum).

Step 12a: **Define variables.**

```
        npks = 2
        suppkn(1) ='Propane'
        suppkn(2) ='LNG'
```

Step 12b: **Call PEAK_SUP and read in the propane data.**

```
        call peak_sup(1)
```

Step 12c: **Call PEAK_SUP and read in the LNG data.**

```
        call peak_sup(2)
```

Note: nesp is the number of additional supply projects.
mxnesp is the maxinum number of extra supply steps.
sname is used to read the scenario name.
supesq is the maxinum quantity that can flow at this price.
supesy is the first year that the project is allowed.
supenm is the name of the supply project.
supesn is the pointer to where the node is located.

Step 13a: **Open 'oth_sup.spc'. 'oth_sup.spc' contains supply specifications for LNG, ANGST, and other supply projects with unique economic factors and constraints. Read in variables.**

```
        nesp=0
        open(19,file='oth_sup.spc')
700     read(19,701,end=790) sname,nname,tesy,tesp,tesq
701     format(2a20,i4,1x,f6.2,1x,f7.1)
        if(tesq.le.0.0) go to 700
        nesp=nesp+1
        if(nesp.gt.mxnesp) call errmsg(4,125)
        supesp(nesp)=tesp
        supesq(nesp)=tesq*1000.0/365.0
        supesy(nesp)=tesy
        supenm(nesp)=sname
        do 710 n=1,nnde
        if(nname.ne.nndnme(n)) go to 710
        supesn(nesp)=n
        go to 700
710     continue
        write(*,711) nname
711     format(' supply project node: ',a20)
        call errmsg(4,126)
```

Step 13b: Close ‘oth_sup.spc’.

```
790     close(19)
        return
        end
```

SUBROUTINES FOUND IN INTRDT.FOR

SUBROUTINE: PEAK_SUP (P1)

CALLED BY: INTRDT.FOR (reads in the transportation and other non-demand data for the LP)

CALLS: ERRMSG.FOR (prints out error and warning messages)

READS: **'**lng.spc' (contains the capacities and costs associated with lng by region)
'propane.spc' (contains the capacities and costs asscociated with propane by region)

CREATES: None

MAIN THEME: Reads in peak supply data (P1 =1 for propane/air, P1 =2 for LNG).

Note: P1 =1 propane
P1 =2 LNG
oldpct is the old peaking percentage to be applied to deliverability.
pkd is the maximum deliverability.
pksc is the storage capacity.
pkfyr is the first year available.
pklc is levelized investment cost.
pklbnd is the peak lower bound for the LP.
pkvc is the variable cost.

Step 1a: **Open either 'propane.spc' or 'lng.spc' (dependent upon p1 flag) and read propane/LNG capacities and costs by region.**

```
if (p1.eq.1) then
open(unit=17,file='propane.spc')
else
open(17,file='lng.spc')
endif

write(99,*)'opening file:'
if (p1.eq.1) then
write(99,*)'propane.spc'
else
write(99,*) 'lng.spc'
endif
```

Step 1b: **Read headers and data from files.**

```
read(17,*) dummy
read(17,*) dummy
```

```
        read(17,*) dummy
        read(17,*) dummy
        read(17,*) dummy
        read(17,*) dummy

665     read(17,*,end=680) nname,tpkd1,tpkst1,tpkfyr1,tpklc1,tpkvc1,
*       tpkfc1,tpkd2,tpkst2,tpkfyr2,tpklc2,tpkvc2,tpkfc2,oldpct
        oldpct = oldpct/100.0
```

Note: oldpct is the percentage to be applied to deliverability for lower bound on usage in the LP (corresponding to a choice of peaking supply not related to economic reasons).

```
        write(99,*)nname,tpkd1,tpkst1,tpkfyr1,tpklc1,tpkvc1,
*       tpkfc1,tpkd2,tpkst2,tpkfyr2,tpklc2,tpkvc2,tpkfc2,oldpct
```

Step 1c: Find the pointer to the current node name.

```
670     format(a20,2(I4,3x,I5,3x,I4,3x,f5.2,3x,f4.2,3x,f4.2,3x))
        do r=1,ndrg
        n=dmnpnt(r)
        if(nname.eq.nndnme(n)) go to 675
        enddo

        write(*,*) nname,fname
672     format(' Mismatch with Node Name' ,a20,' file:')

        call errmsg(3,204)
        stop
        return

675     continue
```

Step 1d: Assign peak supply data to common block variables to be used by other programs.

```
        pkd  (p1,n,1) = float(tpkd1)
        pksc (p1,n,1) = float(tpkst1)
        pkfyr(p1,n,1) = tpk1fyr1
        pklc (p1,n,1) = tpklc1
        pklbnd(p1,n)  = oldpct*pkd(p1,n,1)

        if (tpkvc1.le.0.0) then
        pkvc(p1,n,1)  = machep
        else
        pkvc (p1,n,1) = tpkvc1
        endif
        pkfc (p1,n,1) = tpkfc1


        pkd  (p1,n,2) = float(tpkd2)
        pksc (p1,n,2) = float(tpkst2)
        pkfyr(p1,n,2) = tpk1fyr2
        pklc (p1,n,2) = tpklc2
        if (tpkvc2.le.0.0) then
        pkvc(p1,n,2) = machep
        else
        pkvc (p1,n,2) = tpkvc2
        endif
        pkfc (p1,n,2) = tpkfc2

        go to 665
```

Step 1e: Close ‘propane.spc’ or ‘lng.spc’.

```
680     close(17)
        return
        end
```

SUBROUTINE: INTRPD.FOR

CALLED BY:	INTRPT.FOR (produces output files)
CALLS:	None
READS:	**‘**pflag.spc’ (is the flag for printing certain intermediate inputs)
CREATES:	**‘**gsamsln.rpt’ (transportation flow summary report output file by region, season, and year)
MAIN THEME:	INTRPD.FOR writes out detailed Transportation Reports.

Note:

nlnk is the number of transport links.
ntme is the number of time periods.
mxntad is the maxinum number of transport additions allowed.
lnkcap is the capacity of the link in mmcf/day (current when m = 1, and new when m > 1).
nndnme is the node name.
nlds is the number of load segments.
ldsdy is the number of days in each load segment.
ndsex is the storage extraction summed over t years.
nsto is the number of storage reservoirs at node n at time t.
nsup is the number of supply increments.
ndin in the number of demand increments.
supldf is the load factors for supply.
npks is the number of peak supply options.
nesp is the number of additional supply projects (frontiers type or LNG).
supenm is the name of the supply project.
ndrg is the number of demand regions.
trncap is the transportation capacity.
lnkfyr is the first year that capacity is available.
tavl is the transportation additions value, read from LP output.

Step 1a: **Open ‘pflag.spc’ and read in value for print flag. ‘pflag.spc’ is the flag for printing certain intermediate outputs (print = 1, do not print = 0). Close file.**

```
        Print flag option (1=print, 0=don't print)
        open(unit=89,file='pflag.spc')
        rewind(89)
        read(89,1) pflag ! E&P print flag option
        read(89,1) pflag ! D&I print flag option
1       format(I1)
        close(89)
        if (pflag.eq.0) go to 2300
```

Step 1b: Write out header of report.

```
        write(21,501) (tme(t),t=ibegyr, iendyr)
501     format(/' Transportation Capacity Summary Report (mmcf/day)'/
*       ' Origin',t21,' Destination',t42,33(3x,i4,2x))
        write(21,502)
502     format(' ')
```

Step 1c: Initialize transportation capacity values to zero.

```
        do 550 q=1,nlnk
        do 510 t=ibegyr, iendyr ! csg was 1,ntme
        trncap(q,t)=0.0
510     continue

        do 540 t=ibegyr, iendyr ! cmg was 1,ntme
        do 530 m=1,mxntad
```

Step 1d: Calculate transportation capacities for each time period (year), link, and region, and write out to the report.

```
        if ((tme(t).eq.tme(ibegyr)).or.  ! first year of study
*       (m.ne.1).or.                ! new capacity
*       (lnkfyr(1,q).ne.0)) then    ! existing capacity later than  1990
        if(lnkfyr(m,q).le.tme(t)) then
        if(m.eq.1) then
        tcap=lnkcap(m,q)
        else
        tcap=amax1(0.0,lnkcap(m,q)-lnkcap(m-1,q))
        endif

        if(tcap.gt.0.0) then
        do 520 t1=t,iendyr ! cmg was t,ntme
        trncap(q,t1)=trncap(q,t1)+tavl(q,t,m)
520     continue
        endif  ! tcap
        endif    ! lnkfyr
        endif
530     continue
540     continue

        o=lnkpnt(1,q)
        d=lnkpnt(2,q)
        write(21,541) nndnme(o),nndnme(d),(trncap(q,t),t=ibegyr,iendyr)
541     format(' ',a20,t22,a20,t42,33(1x,f8.2))
550     continue
```

Step 1e: For each gas season, year, and region, write out header for report and calculate flows and percent utilization by link and year.

Note:	tnfvl is the transportation forward flow, read from LP.
	tnrvl is the transportation reverse flow, read from LP.
	trnflw is the net flow.

trnfla is the total flow.
trnutl is the maximum utilization (%).

```
        do 660 l=1,nlds
        write(21,602)
602     format(/' --------------------------------------------')
        write(21,601) l,ldsdy(l),(tme(t),t=ibegyr,iendyr)
601     format(/' Trans. Flow Report for L-Period: ',i2,' Days: ',i3,
1       ' (mmcf/day)'/
2       ' Origin',t21,' Destination',t42,33(3x,i4,2x))
        write(21,502)
        do 650 q=1,nlnk
        do 640 t=ibegyr, iendyr ! cmg was 1,ntme
        trnflw(q,t)=tnfvl(q,t,l)-tnrvl(q,t,l)
        if(l.eq.1) then
        trnfla(q,t)=0.0
        trnutl(q,t)=0.0
        endif
        trnfla(q,t)=trnfla(q,t)+trnflw(q,t)*ldsdy(l)/1000.
        if(trncap(q,t).gt.0.0) then

        trnutl(q,t)=amax1(trnutl(q,t),
1       abs(trnflw(q,t)/trncap(q,t))*100.0)
        endif
640     continue
        o=lnkpnt(1,q)
        d=lnkpnt(2,q)
        write(21,541) nndnme(o),nndnme(d),(trnflw(q,t),t=ibegyr,iendyr)
650     continue
660     continue
```

Step 1f: Produce Transportation Flow Summary Report for each year and region.

```
        write(21,602)
        write(21,701) (tme(t),t=ibegyr,iendyr)
701     format(/' Trans. Flow Report - Annual (Bcf)'/
1       ' Origin',t21,' Destination',t42,33(3x,i4,2x))
        write(21,502)
        do 750 q=1,nlnk
        o=lnkpnt(1,q)
        d=lnkpnt(2,q)
        write(21,541) nndnme(o),nndnme(d),(trnfla(q,t),t=ibegyr,iendyr)
750     continue
```

Step 1g: Produce transportation flow summary report for each year and region - maximum utilization.

```
        write(21,602)
        write(21,751) (tme(t),t=ibegyr,iendyr)  ! cmg was 1,ntme
751     format(/' Trans. Flow Report - Maximum Utilization (%)'/
1       ' Origin',t21,' Destination',t42,33(3x,i4,2x))
        write(21,502)
        do 760 q=1,nlnk
        o=lnkpnt(1,q)
        d=lnkpnt(2,q)
        write(21,541) nndnme(o),nndnme(d),(trnutl(q,t),t=ibegyr,iendyr)
760     continue
```

Step 1h: Produce node flow reports for each region, season, and year.

```
        do 1500 n=1,nnde
        write(21,602)
        write(21,801) nndnme(n)
801     format(' Node Flow Report for : ',a20)

        do 1490 l=1,nlds+1
        if(l.le.nlds) then
        write(21,802) l,ldsdy(l)
802     format(/' Load Period: ',i2,' Number of Days:',i3,
1       ' (mmcf/day)')
        else
        write(21,803)
803     format(/' Annual Averages (Bcf)')
        endif

        write(21,804) (tme(t),t=ibegyr,iendyr)
804     format(' ',t42,33(3x,i4,2x))
```

Step 1i: Intialize total and sub-total arrays.

```
        do 810 t=ibegyr,iendyr ! cmg was 1,ntme
        ndint(t)    = 0.0
        ndifu(t)    = 0.0
        ndnti(t)    = 0.0
        ndsex(t)    = 0.0
        ndsup(t)    = 0.0
        ndpks(t)    = 0.0
        ndosp(t)    = 0.0
        ndnsp(t)    = 0.0
        nditr(t)    = 0.0
        ndtsp(t)    = 0.0
        ndout(t)    = 0.0
        ndsin(t)    = 0.0
        nddmn(t)    = 0.0
        nddmt(t)    = 0.0
810     continue
```

Step 1j: Report flows into nodes.

```
        write(21,811)
811     format(' Transport to Node from Specified Location')
        do 830 q=1,nlnk
        o=lnkpnt(1,q)
        d=lnkpnt(2,q)
        if (d.eq.n) then
        nname=nndnme(o)
        elseif (o.eq.n) then
        nname=nndnme(d)
        endif
        if ((o.eq.n).or.(d.eq.n)) then
        do 825 t=ibegyr,iendyr ! cmg was 1,ntme
        if(d.eq.n) then
        if(l.le.nlds) then
        ndin(t)=tnfvl(q,t,l)
        else
        ndin(t)=0.0
        do 815 l1=1,nlds
        ndin(t)=ndin(t)+tnfvl(q,t,l1)*ldsdy(l1)/1000.
815     continue
        endif
        else
        if(l.le.nlds) then
        ndin(t)=tnrvl(q,t,l)
        else
        ndin(t)=0.0
        do 820 l1=1,nlds
        ndin(t)=ndin(t)+tnrvl(q,t,l1)*ldsdy(l1)/1000.
820     continue
        endif
        endif
        ndint(t)=ndint(t)+ndin(t)
825     continue
        write(21,826) nname,(ndin(t),t=ibegyr,iendyr)
826     format(' ',a20,t42,33(1x,f8.2))
        endif
830     continue
        write(21,831) (ndint(t),t=ibegyr,iendyr)
831     format(' Total Flows In',t42,33(1x,f8.2))
```

Step 1k: Report fuel use on flows into nodes.

```
        write(21,832)
832     format(/' Fuel Use on Transport to Node')
        do 850 q=1,nlnk
        o=lnkpnt(1,q)
        d=lnkpnt(2,q)
        if (d.eq.n) then
        nname=nndnme(o)
        elseif (o.eq.n) then
        nname=nndnme(d)
        endif
        val=lnkfus(q)/100.0
        if ((o.eq.n).or.(d.eq.n)) then
        do 845 t=ibegyr,iendyr ! cmg was 1,ntme
        if(d.eq.n) then
        if(l.le.nlds) then
        ndin(t)=tnfvl(q,t,l)*val
        else
        ndin(t)=0.0
        do 835 l1=1,nlds
        ndin(t)=ndin(t)+tnfvl(q,t,l1)*ldsdy(l1)/1000.*val
835     continue
        endif
        else
        if(l.le.nlds) then
        ndin(t)=tnrvl(q,t,l)*val
        else
        ndin(t)=0.0
        do 840 l1=1,nlds
        ndin(t)=ndin(t)+tnrvl(q,t,l1)*ldsdy(l1)/1000.*val
840     continue
        endif
        endif
        ndifu(t)=ndifu(t)+ndin(t)
845     continue
```

```
        write(21,826) nname,(ndin(t),t=ibegyr,iendyr)
        endif
850     continue
        write(21,851) (ndifu(t),t=ibegyr,iendyr) ! cmg was 1,ntme
851     format(' Fuel Use In Transportation',t42,33(1x,f8.2))
```

Step 1l: Report net flows (less fuel use) into nodes.

```
        do 860 t=ibegyr,iendyr
        ndnti(t)=ndint(t)-ndifu(t)
860     continue
        write(21,861) (ndnti(t),t=ibegyr,iendyr)
861     format(/' Net Transport to Node',t42,33(1x,f8.2))
```

Step 1m: Report storage extraction.

Note: l is the gas load segment index (l=1,nlds)=> MMCF/day data. (l=nlds+1)=> BCF/year data.

sevl = storage extraction value, read from LP (MMCF/day)
ndsex= storage extraction summed over t (years)

```
        write(21,862)
862     format(/' Storage Extraction')
```

Step 1n: Calculate the total storage extraction (summed over reservoirs) by year and season (as well as annual summary) for each node.

```
        do t=ibegyr,iendyr
        if (nsto(n,t).gt.0) then

        do v = 1,nsto(n,t)

        if(l.le.nlds) then        ! in MMCF/day
        ndin(t)=sevl(n,t,v,l)
        else
        ndin(t)=0.0             ! in BCF/year
        do  l1=1,nlds
        ndin(t)=ndin(t)+sevl(n,t,v,l1)*ldsdy(l1)/1000.0
        enddo ! l1 loop
        endif
        ndsex(t)=ndsex(t)+ndin(t)
        enddo    ! v loop
        endif
        enddo        ! t loop
```

Step 1o: Write out storage extraction values.

```
        write(21,881) (ndsex(t),t=ibegyr,iendyr)
881     format(' Total Storage Extraction: ',t42,33(1x,f8.2))
```

Step 1p: Calculate and report basic supply.

```
        ifl=0
        do 920 k=1,nsup
        do 915 s=1,nsrg
```

```
          if(suppnt(s).eq.n) then   !if current node n is a supply node
          ifl=1
          do 912 t1=ibegyr,iendyr
          if (t1.eq.ibegyr) then
          vleft=1.0
          else
          vleft=1.0-vdr**(tme(t1)-tme(t1-1))
          endif

          do 910 t=t1,iendyr
          vdo=vleft*vdr**(tme(t)-tme(t1))
          if (l.le.nlds) then
          ndsup(t)=ndsup(t)+suptot(t,s,k)*
1         supldf(s,l)*spvl(k,t1)*vdo
          else
          ndsup(t)=ndsup(t)+suptot(t,s,k)*
1         365./1000.*spvl(k,t1)*vdo
          endif
910       continue
912       continue
          endif
915       continue
920       continue

          if(ifl.eq.1) then
          write(21,921) (ndsup(t),t=ibegyr,iendyr)
921       format(/' Nat. Gas Supply: ',t42,33(1x,f8.2))
          endif
```

Step 1q: Calculate and report peak supply.

```
          ifl=0
          do 940 p=1,npks
          do 935 r=1,ndrg
          if(dmnpnt(r).eq.n) then   ! if current node n is a demand node
          do 930 t=ibegyr,iendyr

          if(l.le.nlds) then     ! in MMCF/day
          olduse(t) =pkovl(p,n,t,l,1)
          olddel(t) =pkd(p,n,1)
          newuse(t) =pkovl(p,n,t,l,2)
          newdel(t) =pkd(p,n,2)
          ndin(t)   =olduse(t)+newuse(t)    ! existing + new

          else                  ! in BCF/year
          ndin(t)=0.0
          olduse(t) =0.0
          newuse(t) =0.0
          do l1=1,nlds
          olduse(t) =olduse(t) +
*         pkovl(p,n,t,l1,1)*ldsdy(l1)/1000.
          newuse(t) =newuse(t) +
*         pkovl(p,n,t,l1,2)*ldsdy(l1)/1000.
          enddo  !l1 loop
          ndin(t)=ndin(t)+ olduse(t) + newuse(t)
          olddel(t) =pksc(p,n,1)/1000.  !for annual numbers use storage capacity
          newdel(t) =pksc(p,n,2)/1000.  !for annual numbers use storage capacity
          endif
          ndpks(t)=ndpks(t)+ndin(t)
930       continue
```

```
        if(ifl.eq.0) then
        write(21,922)
922     format(/' Peaking Supply Sources:')
        ifl=1
        endif

        write(21,*)' '
        write(21,826) suppkn(p)
        write(21,945) (olduse(t),t=ibegyr,iendyr)
        write(21,946) (olddel(t),t=ibegyr,iendyr)
        write(21,*)' '
        write(21,947) (newuse(t),t=ibegyr,iendyr)
        write(21,948) (newdel(t),t=ibegyr,iendyr)
        write(21,*)' '
        write(21,949) (ndin(t),t=ibegyr,iendyr)
        write(21,950) (olddel(t)+newdel(t),t=ibegyr,iendyr)

        endif
935     continue
940     continue
        if(ifl.eq.1) then
        write(21,941) (ndpks(t),t=ibegyr,iendyr)
941     format(/' Total Peaking Supply Usage: ',t42,33(1x,f8.2))
        endif
945     format(' ','Existing  ',1x,'Usage      ',t42,33(1x,f8.2))
946     format(' ','Existing  ',1x,'Maximum ',   t42,33(1x,f8.2))
947     format(' ','New       ',1x,'Usage      ',t42,33(1x,f8.2))
948     format(' ','New       ',1x,'Maximum ',   t42,33(1x,f8.2))
949     format(' ','Total     ',1x,'Usage      ',t42,33(1x,f8.2))
950     format(' ','Total     ',1x,'Maximum ',   t42,33(1x,f8.2))
```

Step 1r: Calculate and report supply from other supply projects.

```
        ifl=0
        do 960 e=1,nesp
        if(supesn(e).eq.n) then
        do 955 t=ibegyr,iendyr
        if (l.le.nlds) then
        ndin(t)=esvl(e,t)
        else
        ndin(t)=esvl(e,t)*365./1000.
        endif
        ndosp(t)=ndosp(t)+ndin(t)
955     continue
        if (ifl.eq.0) then
        write(21,942)
942     format(/' Other Supply Source')
        ifl=1
        endif
        write(21,826) supenm(e),(ndin(t),t=ibegyr,iendyr)
        endif
960     continue
        if(ifl.eq.1) then
        write(21,961) (ndosp(t),t=ibegyr,iendyr)
961     format(' Total Other Supply: ',t42,33(1x,f8.2))
        endif
```

Step 1s: Calculate net supply.

```
        do 970 t=ibegyr,iendyr
        ndnsp(t)=ndnti(t)+ndsex(t)+ndsup(t)+ndpks(t)+ndosp(t)
970     continue
```

Step 1t: Calculate and report level of demand interruption.

```
        ifl=0
        do 990 r=1,ndrg
        if(dmnpnt(r).eq.n) then
        ifl=1
        do 986 t=ibegyr,iendyr
        if(l.le.nlds) then
        do 971 j=1,4
        nditr(t)=nditr(t)+ifdmi(r,t,l,j)
971     continue
        do 973 f=1,7
        if (f.gt.3) then
        nditr(t)=nditr(t)+euofl(r,t,l,f)
        endif
973     continue
        else
        do 985 l1=1,nlds
        do 981 j=1,4
        nditr(t)=nditr(t)+ifdmi(r,t,l1,j)*ldsdy(l1)/1000.0
981     continue
        do 983 f=1,7
        if (f.gt.3) then
        nditr(t)=nditr(t)+euofl(r,t,l1,f)*ldsdy(l1)/1000.0
        endif
983     continue
985     continue
        endif
986     continue
        endif
990     continue
        write(21,992)
992     format(' ')

        if(ifl.eq.1) then
        write(21,991) (nditr(t),t=ibegyr,iendyr)

991     format(' Total Interuption: ',t42,33(1x,f8.2))
        endif
```

Step 1u: Calculate and report total supply (with interruption added).

```
        do 995 t=ibegyr,iendyr
        ndtsp(t)=ndnsp(t)+nditr(t)
995     continue
        write(21,996) (ndtsp(t),t=ibegyr,iendyr)
996     format(' Total Supply: ',t42,33(1x,f8.2))
```

Step 1v: Calculate and report net transport from node.

```
        write(21,1002)
1002    format(/' Transport from Node to Specified Location')
        do 1020 q=1,nlnk
        o=lnkpnt(1,q)
        d=lnkpnt(2,q)
        if(d.eq.n) then
        nname=nndnme(o)
        elseif(o.eq.n) then
        nname=nndnme(d)
        endif
        if((o.eq.n).or.(d.eq.n)) then
        do 1015 t=ibegyr,iendyr
        if(d.eq.n) then
        if(l.le.nlds) then
        ndin(t)=tnrvl(q,t,l)
        else
        ndin(t)=0.0
        do 1005 l1=1,nlds
        ndin(t)=ndin(t)+tnrvl(q,t,l1)*ldsdy(l1)/1000.
1005    continue
        endif
        else
        if(l.le.nlds) then
        ndin(t)=tnfvl(q,t,l)
        else
        ndin(t)=0.0
        do 1010 l1=1,nlds
        ndin(t)=ndin(t)+tnfvl(q,t,l1)*ldsdy(l1)/1000.
1010    continue
        endif
        endif
        ndout(t)=ndout(t)+ndin(t)
1015    continue
        write(21,826) nname,(ndin(t),t=ibegyr,iendyr)
        endif
1020    continue
        write(21,1021) (ndout(t),t=ibegyr,iendyr)

1021    format(' Total Flows Out',t42,33(1x,f8.2))
```

Step 1w: **Calculate and report storage injection at node.**

Note: l is the gas load segment index (l=1,nlds)=> MMCF/day data. (l=nlds+1)=> BCF/year data.

sivl is the storage injection value, read from LP (MMCF/day).
ndsin is the storage injection summed over t (years).

```
        write(21,1022)
1022    format(/' Storage Injection')
```

Step 1x: **Calculate and report total storage injections (summed over reservoirs) by year and season (as well as annual summary) for node.**

```
        do t=ibegyr,iendyr
        if (nsto(n,t).gt.0) then

        do v = 1,nsto(n,t)

        if(l.le.nlds) then       ! in MMCF/day
        ndin(t)=sivl(n,t,v,l)
        else
        ndin(t)=0.0             ! in BCF/year
        do  l1=1,nlds
        ndin(t)=ndin(t)+sivl(n,t,v,l1)*ldsdy(l1)/1000.0
        enddo ! l1 loop
        endif
        ndsin(t)=ndsin(t)+ndin(t)
        enddo    ! v loop
        endif
        enddo  ! t loop

        write(21,1041) (ndsin(t),t=ibegyr,iendyr)

1041    format(' Total Storage Injection: ',t42,33(1x,f8.2))
```

Step 1y: Calculate Total Demand (Firm and Interruptable: Demand Regions)

```
        ifl=0
        do r=1,ndrg
        if (dmnpnt(r).eq.n) then  ! if current node n is a demand node
        ifl=1
        do t=ibegyr,iendyr
        if (l.le.nlds) then
        nddmn(t)=nddmn(t)+rsdmn(r,t,l)+cmdmn(r,t,l)+ifdmn(r,t,l)
        do j=1,4
        nddmn(t)=nddmn(t)+ifdmi(r,t,l,j)
        enddo ! j loop

        nddmn(t)=nddmn(t)+eugas(r,t,l)
        do f=1,7
        if(f.gt.3) then
        nddmn(t)=nddmn(t)+euofl(r,t,l,f)
        endif
        enddo ! f loop
        else
        do l1=1,nlds
        nddmn(t)=nddmn(t)+(rsdmn(r,t,l1)+cmdmn(r,t,l1)+
*       ifdmn(r,t,l1))*(ldsdy(l1)/1000.0)
        do j=1,4
        nddmn(t)=nddmn(t)+ifdmi(r,t,l1,j)*ldsdy(l1)/1000.0
        enddo ! j loop

        nddmn(t)=nddmn(t)+eugas(r,t,l1)*(ldsdy(l1)/1000.0)
        do f=1,7
        if(f.gt.3) then
        nddmn(t)=nddmn(t)+euofl(r,t,l1,f)*(ldsdy(l1)/1000.0)
        endif
        enddo ! f  loop
        enddo   ! l1 loop
        endif     ! if (l.le.nlds)
        enddo       ! t loop
        endif         ! if (dmnpnt(r).eq.n)
        enddo           ! r loop
        write(21,1091) (nddmn(t),t=ibegyr,iendyr)

1091    format(/' Customer Demand: ',t42,33(1x,f8.2))
```

Step 1z: Calculate and report lease and plant usage for the supply nodes.

```
        do t=ibegyr,iendyr
        if (l.le.nlds) then
        nlsepln(t,l)=ndsup(t)*lsepln/100.0   !in MMCF/day
        else
        nlseplnt(t) = 0.0
        do l1=1,nlds
        nlseplnt(t)=nlseplnt(t)+nlsepln(t,l1)*
*       (ldsdy(l1)/1000.0)     !in BCF
        enddo   ! l1 loop
        endif    ! if (l.le.nlds)
        nddo      ! t loop

        if (l.le.nlds) then
        write(21,1092) (nlsepln(t,l),t=ibegyr,iendyr)
        else
        write(21,1092) (nlseplnt(t),t=ibegyr,iendyr)
        end if
1092    format(/' Lease and Plant Usage: ',t42,33(1x,f8.2))
```

Step 1aa: Calculate and report total demand and total supply by node, season, and time period.

```
        do t=ibegyr,iendyr
        if (l.le.nlds) then
        nddmt(t)=nddmn(t)+ndsin(t)+ndout(t)+nlsepln(t,l)
        else
        nddmt(t)=nddmn(t)+ndsin(t)+ndout(t)+nlseplnt(t)
        end if
        enddo  ! t loop

        write(21,1111) (nddmt(t),t=ibegyr,iendyr)

1111    format(' Total Demand: ',t42,33(1x,f8.2))

        write(21,*)' '
        write(21,*)'**************************************************'
        write(21,1112)(ndtsp(t)-nddmt(t),t=ibegyr,iendyr)
        write(21,*)'**************************************************'
        write(21,*)' '
1112    format(' Total Supply-Total Demand:',t42,33(1x,f8.2))

1490    continue        ! load loop l
1500    continue         ! node loop n


2300    return
        end
```

SUBROUTINE: INTRPG.FOR

CALLED BY: INTRPT.FOR (produces output files)
INTRVS.FOR (check for convergence and generate gasprc.new)

CALLS: CHAR2NUM.FOR (converts base 62 numbers to standard base 10)
NUM2CHAR.FOR (converts standard base 10 numbers to base 62)
RGET.FOR (reads the records from the LP solution file)
ZAP.FOR (zeros out an array)

READS: **‘**dual_prc.spc’ (contains the dual price of natural gas according to season and region as compiled in the LP and adjusted for present value)
‘gasall.prt’ (is the LP solution file)

CREATES: None

MAIN THEME: This subroutine reads results from the linear program solution file.

Note: vcde is the base 62 version of the storage reservoir index (v).
(see program NUM2CHAR.FOR for details)

```
integer     v1,v2,j1,j2
Character*2 vcde,jcde

Data cde/'0','1','2','3','4','5','6','7','8','9',
*        'A','B','C','D','E','F','G','H','I','J',
*        'K','L','M','N','O','P','Q','R','S','T',
*        'U','V','W','X','Y','Z','a','b','c','d',
*        'e','f','g','h','i','j','k','l','m','n',
*        'o','p','q','r','s','t','u','v','w','x',
*        'y','z'/
Data pscale/0.2/
Data lsepln/6.5/
```

Step 1a: **Open Solution File, ‘gasall.prt’, and skip file headers. ‘gasall.prt’ is the LP solution file.**

```
Open(14,file='gasall.prt')
10 read(14,11) v8
11 format(t4,a8)
if(v8.ne.'Number  ') go to 10

call rget('OBJ     ',oval,odual,0,0,0,0,0,0,cde,mxc)
```

Step 1b: **Zero out row dual value arrays by calling ZAP.FOR.**

Note: mxnnde is the maxinum number of nodes.
mxnlnk is the maxinum number of links.
mxntme is the maxinum number of time periods.
mxnlds is the maxinum number of load segments.
mxnres is the maxinum number of storage reservoirs for a fixed node, and year.
mxntad is the maxinum number of transport additions allowed per link.
mxnesp is the maxinum number of extra supply steps (e.g. frontiers, LNG).
mxnpks is the maxinum number of peak supply types.

```
        call zap(mddl,mxnnde,mxntme,mxnlds,1,1)
        call zap(tcdl,mxnlnk,mxntme,mxnlds,1,1)
        call zap(txdl,mxnlnk,mxntad,1,1,1)
        call zap(skdl,1,1,1,1,1)
        call zap(scdl,mxnnde,mxntme,mxnres,1,1)
        call zap(sddl,mxnnde,mxntme,mxnres,1,1)
        call zap(sxdl,mxnnde,mxntme,mxnres,1,1)
        call zap(sedl,mxnnde,mxntme,mxnres,1,1)
        call zap(eadl,mxnnde,mxntme,mxnres,1,1)
        call zap(ebdl,mxnnde,mxntme,mxnres,1,1)
        call zap(ccadl,mxntme,1,1,1,1)
        call zap(ccddl,mxntme,mxnlds,1,1,1)
        call zap(esdl,mxnesp,1,1,1,1)
        call zap(pkdl,mxnpks,mxnnde,mxntme,2,1)
        call zap(pksdl,mxnpks,mxnnde,1,2,1)
```

Step 1c: **Read in material balance constraint values.**

```
200     do 250 n=1,nnde
        do 240 t=1,ntme
        do 230 l=1,nlds
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        call rget('MB###   ',valx,mddl(n,t,l),n,t,l,0,0,0,cde,mxc)
        end if
230     continue
240     continue
250     continue
```

Step 1d: **Read in transport capacity constraint values.**

Note: nlnk is the number of transport links.
ntme is the number of time periods.
RGET.FOR reads records from the matrix solution file.

```
        do 350 q=1,nlnk
        q1=(q-1)/mxc+1
        q2=q-(q1-1)*mxc
        do 340 t=1,ntme
        do 330 l=1,nlds
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        call rget('TC####  ',valx,tcdl(q,t,l),q1,q2,t,l,0,0,cde,mxc)
        end if
330     continue
340     continue
350     continue
```

Step 1e: **Read in transport capacity addition convexity constraint values.**

Note: lnkfyr is the first year that capacity is available.

```
        do 450 q=1,nlnk
        q1=(q-1)/mxc+1
        q2=q-(q1-1)*mxc
        do 440 m=1,mxntad
        if(lnkcap(m,q).gt.0.0) then
        if((m.gt.1).or.(lnkfyr(m,q).ne.0)) then
        call rget('TX###   ',valx,txdl(q,m),q1,q2,m,0,0,0,cde,mxc)
        endif
        endif
440     continue
450     continue
```

Step 1f: **Read in supply convexity constraint values.**

Note: nsup is the number of supply increments.
suppas is the pass of supply increment.
nsrg is the number of supply regions.
suptot is the supply for the load segment at the specified price(mmcf/day).

```
        vscls=0.0
        nscl=0
        do 490 k=1,nsup
        ulmt=max(0.0,1.0-pscale*(nsps-suppas(k)))
        if(ulmt.gt.0.0) then
        do 480 s=1,nsrg
        do 470 t=1,ntme
        if(suptot(t,s,k).gt.0.0) then
        vscls=vscls+suptot(t,s,k)*(1.0-lsepln/100.0)
        nscl=nscl+1
        endif
470     continue
480     continue
        endif
490     continue
        if((nscl.gt.1).and.(vscls.ne.0.0)) then
        vscls=vscls/float(nscl)
        else
        vscls=1.0
        endif

        do 495 t=1,ntme
```

```
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          call rget('SK#     ',valx,skdl,t,0,0,0,0,0,cde,mxc)
          end if
495       continue
```

Note: strvcp is the maxinum volume capacity of storage in mmcf.

Step 2a: Read in storage volume constraint, storage capacity constraint, and storage extraction constraint values.

```
          do n=1,nnde
          do t=1,ntme
          if(nsto(n,t).gt.0) then
          do v=1,nsto(n,t)

          call num2char(v,vcde)    ! convert decimal v to base 62 vcde
          call char2num(vcde,v1,v2)! get decimal digits v1 and v2
          if (strvcp(n,t,v).gt.0.0) then
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          call rget('SC####  ',valx,scdl(n,t,v),
1         n,t,v1,v2,0,0,cde,mxc)
          call rget('SD####  ',valx,sddl(n,t,v),
1         n,t,v1,v2,0,0,cde,mxc)
          call rget('SX####  ',valx,sxdl(n,t,v),
1         n,t,v1,v2,0,0,cde,mxc)
          call rget('EA####  ',valx,eadl(n,t,v),
1         n,t,v1,v2,0,0,cde,mxc)
          call rget('EB####  ',valx,ebdl(n,t,v),
1         n,t,v1,v2,0,0,cde,mxc)
          end if
          end if
          enddo    ! v loop
          endif
          enddo        ! t loop
          enddo          ! n loop
```

Step 2b: Read in demand convexity constraint values.

```
          do 750 r=1,ndrg
          n=dmnpnt(r)
          do 710 t=1,ntme
          do 705 z=1,4
          do 704 f=1,7
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          call rget('DX####  ',valx,valy,n,t,z,f,0,0,cde,mxc)
          end if
704       continue
705       continue
710       continue
          do 730 t=1,ntme
          do 725 z=1,4
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          call rget('DY###   ',valx,valy,n,t,z,0,0,0,cde,mxc)
          end if
725       continue
730       continue
750       continue
```

Step 3a: **Set cost accumulation rows - and convert mddl to annual (undiscounted) dollars per mcf. Write the dual prices to 'dual_prc.spc' by region, year, and season.**

```
        open(unit=29,file='dual_prc.spc')
        do 850 t=1,ntme
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        call rget('CCA#    ',valx,ccadl(t),t,0,0,0,0,0,cde,mxc)
        end if
        do 840 l=1,nlds
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        call rget('CCD##   ',valx,ccddl(t,l),t,l,0,0,0,0,cde,mxc)
        end if
        do 830 n=1,nnde
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        end if

        mddl(n,t,l)=-mddl(n,t,l)/ccddl(t,l)
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        write(29,*)n,t,l,mddl(n,t,l)
        end if
830     continue
840     continue
850     continue

        close(29)
```

Step 3b: **Read in extra supply project constraint values.**

Note: nesp is the number of additional supply projects.

```
        do 890 e=1,nesp
        call rget('ES#     ',valx,esdl(e),e,0,0,0,0,0,cde,mxc)
890     continue
```

Step 3c: **Read in peak supply constraint values.**

Note: npks is the number of peak supply options.
ndrg is the number of demand regions.

```
        do p=1,npks
        do r=1,ndrg
        n=dmnpnt(r)
        do k=1,2
        call rget('PKS###  ',valx,pksdl(p,n,k),
*       p,n,k,0,0,0,cde,mxc)
        end do
        do t=1,ntme
        do k=1,2
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        call rget('PK####  ',valx,pkdl(p,n,t,k),
*       p,n,t,k,0,0,cde,mxc)
        end if

        enddo ! k loop
        enddo   ! t loop
        enddo     ! r loop
        enddo       ! p loop
```

Step 3d: Skip to columns section of matrix.

```
        read(14,12) v8
        if(v8.ne.'        ') stop 'bad end to rows section'
        read(14,12) v8
        if(v8.ne.'Columns ') stop 'bad start to columns section'
12      format(t1,a8)
```

Step 3e: Zero out column value arrays.

```
        call zap(tnfvl,mxnlnk,mxntme,mxnlds,1,1)
        call zap(tnrvl,mxnlnk,mxntme,mxnlds,1,1)
        call zap(tavl,mxnlnk,mxntme,mxntad,1,1)

        call zap(svvl,mxnnde,mxntme,mxnres,1,1)
        call zap(scvl,mxnnde,mxntme,mxnres,1,1)
        call zap(sevl,mxnnde,mxntme,mxnres,mxnlds,1)
        call zap(sivl,mxnnde,mxntme,mxnres,mxnlds,1)

        call zap(esvl,mxnesp,mxntme,1,1,1)

        call zap(pkovl,mxnpks,mxnnde,mxntme,mxnlds,2)
        call zap(pkivl,mxnpks,mxnnde,mxntme,2,1)

        call zap(spvl,mxnsin,mxntme,1,1,1)
        call zap(odvl,mxntme,mxnlds,1,1,1)
        call zap(oavl,mxntme,1,1,1,1)
        call zap(euofl,mxndrg,mxntme,mxnlds,7,1)
        call zap(eugas,mxndrg,mxntme,mxnlds,1,1)
        call zap(eucap,mxndrg,mxntme,7,2,1)
```

Step 3f: Read in the transportation flow values.

Note: Transport Activity - forward direction (TNFqtl) and reverse direction (TNRqtl)

```
1000    do 1050 q=1,nlnk
        q1=(q-1)/mxc+1
        q2=q-(q1-1)*mxc
        do 1040 t=1,ntme
        do 1030 l=1,nlds
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        call rget('TNF#### ',tnfvl(q,t,l),valx,q1,q2,t,l,0,0,cde,mxc)
        call rget('TNR#### ',tnrvl(q,t,l),valx,q1,q2,t,l,0,0,cde,mxc)
        end if
1030    continue
1040    continue
1050    continue
```

Step 3g: Read in values for transportation capacity additions.

Note: Transport Capacity Additions (TAqtm)
lnkcap is the capacity of the link.
m = 1, existing
m = 2, new

```
        do 1150 q=1,nlnk
        q1=(q-1)/mxc+1
        q2=q-(q1-1)*mxc
        do 1140 t=1,ntme
        do 1130 m=1,mxntad

        if ((tme(t).eq.tme(ibegyr)).or.   ! first year of study
*       (m.ne.1).or.                ! new capacity
*       (lnkfyr(1,q).ne.0)) then     ! existing capacity later than 1990

        if(lnkfyr(m,q).le.tme(t)) then
        if(m.eq.1) then
        tcap=lnkcap(m,q)
        else
        tcap=amax1(0.0,lnkcap(m,q)-lnkcap(m-1,q))
        endif
        if(tcap.gt.0.0) then
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        call rget('TA####  ',tavl(q,t,m),valx,q1,q2,t,m,0,0,cde,mxc)
        end if
        endif
        endif
        endif
1130    continue
1140    continue
1150    continue
```

Step 3h: Read in residential demand values.

```
        vprlow=  3.30
        vprhgh= 11.00
        vinc  =  0.07

        do 1220 r=1,ndrg
        n=dmnpnt(r)
        do 1215 t=1,ntme
        tqnto=0.0
        tprco=0.0
        do 1210 j=1,((vprhgh-vprlow)/vinc)+1
        tprc=(vprhgh-vinc*float(j-1))+dmsdmr(1,r)
        tqntn=rdmbqn(t,r)*(tprc/rdmbpr(t,r))**rdmpel(r)
        tqnt=amax1((tqntn-tqnto),0.0)
        tqnto=tqntn
        call num2char(j,jcde)     ! convert decimal v to base 62 vcde
        call char2num(jcde,j1,j2) ! get decimal digits v1 and v2
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        call rget('DR####  ',valx,val,n,t,j1,j2,0,0,cde,mxc)
        end if
        do 1205 l=1,nlds
        tqntl=-1.0*(1000.0/365.0)*rdmldf(l,r)
        if(j.eq.1) then
```

```
          rsdmn(r,t,l)=0.0
          endif
          rsdmn(r,t,l)=rsdmn(r,t,l)-tqntl*valx
1205      continue
1210      continue
1215      continue
1220      continue
```

Step 3i: Read in commercial demand values.

```
          vprlow=  2.00
          vprhgh=  9.00
          vinc  =  0.07

          do 1240 r=1,ndrg
          n=dmnpnt(r)
          do 1235 t=1,ntme
          tqnto=0.0
          tprco=0.0
          do 1230 j=1,((vprhgh-vprlow)/vinc)+1
          tprc=(vprhgh-vinc*float(j-1))+dmsdmr(2,r)
          tqntn=cdmbqn(t,r)*(tprc/cdmbpr(t,r))**cdmpel(r)
          tqnt=amax1((tqntn-tqnto),0.0)
          tqnto=tqntn
          call num2char(j,jcde)     ! convert decimal v to base 62 vcde
          call char2num(jcde,j1,j2) ! get decimal digits v1 and v2
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          call rget('DC####  ',valx,val,n,t,j1,j2,0,0,cde,mxc)
          end if
          do 1225 l=1,nlds
          tqntl=-1.0*(1000.0/365.0)*cdmldf(l,r)*(1.0-cdmish(r))
          if(j.eq.1) then
          cmdmn(r,t,l)=0.0
          endif
          cmdmn(r,t,l)=cmdmn(r,t,l)-tqntl*valx
1225      continue
          if(cdmish(r).gt.0.0) then
          do 1228 l=1,nlds
          tqntl=-1.0*(1000.0/365.0)*cdmldf(l,r)*cdmish(r)
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          call rget('DC##### ',valx,val,n,t,j1,j2,l,0,cde,mxc)
          end if
          cmdmn(r,t,l)=cmdmn(r,t,l)-valx*tqntl
1228      continue
          endif
1230      continue
1235      continue
1240      continue
```

Step 3j: Read in industrial demand values.

Note: idmshr is the share of industrial demand that is oil or gas.
idmefc is the demand by time period and industrial subsector.
niss is the number of industrial subsectors.
idmish is the share of industrial subsector demand by region that is interruptable (%).
idmldf is the load profile of industrial demand.

```
      do 1300 r=1,ndrg
      n=dmnpnt(r)
      do 1295 t=1,ntme
      do 1290 j=1,4
      vscl=0.0
      do 1243 ss=1,niss
      tqnt=idmefc(t,r,ss)*(100.0-idmish(r,ss))/100.0*(1000.0/365.0)
      vscl=vscl+tqnt*idmshr(t,r,ss,1)*idmshr(t,r,ss,j+1)
1243  continue
      if(vscl.gt.0.0) then
      if ((t.ge.ibegyr).and.(t.le.iendyr)) then
      call rget('DI###   ',valx,val,n,t,j,0,0,0,cde,mxc)
      end if
      do 1250 l=1,nlds
      tqntl=0.0
      do 1245 ss=1,niss
      tqnt=idmefc(t,r,ss)*(1.0-idmish(r,ss)/100.0)*(1000.0/365.0)
      tqntl=-tqntl+tqnt*idmshr(t,r,ss,1)*idmshr(t,r,ss,j+1)*
*     idmldf(l,r,ss)
1245  continue
      if(vscl.ne.0.0) then
      tqntl=tqntl/vscl
      endif
      if(j.eq.1) then
      ifdmn(r,t,l)=0.0
      endif
      ifdmn(r,t,l)=ifdmn(r,t,l)-tqntl*valx
      ifdmo(r,t,l,j)=-tqntl*(vscl-valx)
1250  continue
      endif
      valy=0.0
      do 1260 l=1,nlds
      tqntl=0.0
      do 1255 ss=1,niss
      tqnt=idmbqn(r,ss)*idmish(r,ss)/100.0*(1000.0/365.0)
      tqntl=tqntl-tqnt*idmshr(t,r,ss,1)*idmshr(t,r,ss,j+1)*
*     idmldf(l,r,ss)
1255  continue
      vscl=-tqntl
      if(vscl.ne.0.0) then
      tqntl=tqntl/vscl
      endif
      if(tqntl.ne.0.0) then
      if ((t.ge.ibegyr).and.(t.le.iendyr)) then
      call rget('DI####  ',valx,val,n,t,j,l,0,0,cde,mxc)
      end if
      tq(l)=-tqntl*vscl
      if(vscl.ne.0.0) then
      tv(l)=valx/vscl
      else
      tv(l)=0.0
      endif
      valy=amax1(valy,tv(l))
      else
      tq(l)=0.0
      tv(l)=0.0
      endif
1260  continue
      do 1270 l=1,nlds
      ifdmn(r,t,l)=ifdmn(r,t,l)+tq(l)*tv(l)
      ifdmo(r,t,l,j)=ifdmo(r,t,l,j)+tq(l)*(1.0-valy)
      ifdmi(r,t,l,j)=tq(l)*(valy-tv(l))
```

```
1270    continue
1290    continue
1295    continue
1300    continue
```

Step 3k: **Read in electric utility demand values.**

Note: euexc is existing generation capacity (10^12 watts-hrs) by fuel type.
euovcf is the load factor for existing capacity.
eunvcf is the load factor specification (fraction) for new capacity.
euflef is the generation efficiency (mcf/000kwh) by fuel type and load.
euldf is the load profile of eu demand by region for each eu load type.

```
        do 1350 r=1,ndrg
        n=dmnpnt(r)
        do 1345 t=1,ntme
        j=0
        do 1335 f=1,7
        do 1330 k=1,2
        j1=j+1
        j=j+4

        valx = 0.0

        if((k.ne.1).or.(t.eq.ibegyr)) then
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        call rget('DE###   ',valx,val,n,t,j1,0,0,0,cde,mxc)
        end if
        else
        valx=0.0
        endif

        do 1315 t1=t,ntme
        if(k.eq.1) then
        if(euexc(r,f,1).gt.0.) then
        val=euexc(r,f,t1)/euexc(r,f,1)
        else
        val=1.0
        endif
        vala=euovcf(r,f,t1)
        else
        val=1.0
        vala=eunvcf(f,1)
        endif

        if(vala.le.0.0) then
        if(valx.gt.0.0) then
        end if
        else
        eucap(r,t1,f,k)=eucap(r,t1,f,k)+
*       valx*val/(8760.0*vala)*1000.0
        endif

1315    continue
```

```
        j=j1-1
        do 1329 z=1,4
        if(k.eq.1) then
        euutlo(r,t,f,z)=0.0
        euutlg(r,t,f,z)=0.0
        endif
        j=j+1
        if((f.ge.1).and.(f.le.3)) then
        l2=nlds+1
        vscl=0.0
        else
        l2=1
        vscl=1.0
        endif
        do 1325 l1=l2,nlds+1
        if(k.eq.1) then
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        call rget('DE####  ',valx,val,n,t,j,l1,0,0,cde,mxc)
        end if
        else
        valx=0.0
        val=0.0
        endif
        do 1320 l=1,nlds
        if(l.lt.l1) then
        tqntl=euflef(f,z,k)*euldf(l,r,z)*(1000.0/365.0)
        euofl(r,t,l,f)=euofl(r,t,l,f)+tqntl*valx
        euutlo(r,t,f,z)=euutlo(r,t,f,z)+valx*ldsdy(l)/365.0
        else
        tqntn=euflef(f,z,k)*euldf(l,r,z)*(1000.0/365.0)
        tqntl=euflef(f,z,k)*euldf(l,r,z)*(1000.0/365.0)*vscl
        eugas(r,t,l)=eugas(r,t,l)+tqntl*valx
        euofl(r,t,l,f)=euofl(r,t,l,f)+(tqntn-tqntl)*valx
        euutlo(r,t,f,z)=euutlo(r,t,f,z)+valx*(1.0-vscl)*
*       ldsdy(l)/365.0
        euutlg(r,t,f,z)=euutlg(r,t,f,z)+valx*vscl*ldsdy(l)/365.0
        endif
1320    continue
1325    continue
1329    continue
1330    continue
1335    continue
1345    continue
1350    continue
```

Step 3l: Read in values for storage volume extraction/injection.

```
        do  n=1,nnde
        do t=1,ntme
        if(nsto(n,t).gt.0) then
        do v=1,nsto(n,t)

        call num2char(v,vcde)     ! convert decimal v to base 62 vcde
        call char2num(vcde,v1,v2) ! get decimal digits v1 and v2
        if (strvcp(n,t,v).gt.0.0) then
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        call rget('SV####  ',svvl(n,t,v),
1       valx,n,t,v1,v2,0,0,cde,mxc)
        end if ! conditional write statement
        end if
        enddo    ! v loop
```

```
          endif
          enddo         ! t loop
          enddo           ! n loop
```

Step 3m: Read in values for storage volume capacity additions.

Note: nsto is the number of reservoirs for year t at node n.
strvcp is the maxinum volume capacity of storage (Mmcf).

```
          do n=1,nnde
          do t=1,ntme
          if(nsto(n,t).gt.0) then
          do v=1,nsto(n,t)

          call num2char(v,vcde)    ! convert decimal v to base 62 vcde
          call char2num(vcde,v1,v2) ! get decimal digits v1 and v2

          if (strvcp(n,t,v).gt.0.0) then
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          call rget('SC####  ',scvl(n,t,v),
1         valx,n,t,v1,v2,0,0,cde,mxc)
          end if
          end if
          enddo  ! v loop
          endif
          enddo        ! t loop
          enddo          ! n loop
```

Step 3n: Read in values for storage extraction rates.

```
          do n=1,nnde
          do t=1,ntme
          if(nsto(n,t).gt.0) then
          do v=1,nsto(n,t)

          call num2char(v,vcde)    ! convert decimal v to base 62 vcde
          call char2num(vcde,v1,v2) ! get decimal digits v1 and v2

          if (strvcp(n,t,v).gt.0.0) then
          do l=1,nlds
          if ((t.ge.ibegyr).and.(t.le.iendyr)) then
          call rget('SE##### ',sevl(n,t,v,l),
1         valx,n,t,v1,v2,l,0,cde,mxc)
          end if
          enddo  ! l loop
          endif
          enddo      ! v loop
          endif
          enddo           ! t loop
          enddo             ! n loop
```

Step 3o: Read in values for storage injection rates.

```
          do n=1,nnde
          do t=1,ntme
          if(nsto(n,t).gt.0) then
          do v=1,nsto(n,t)
```

```
           call num2char(v,vcde)    ! convert decimal v to base 62 vcde
           call char2num(vcde,v1,v2) ! get decimal digits v1 and v2

           if (strvcp(n,t,v).gt.0.0) then
           do  l=1,nlds
           if ((t.ge.ibegyr).and.(t.le.iendyr)) then
           call rget('SI##### ',sivl(n,t,v,l),
1          valx,n,t,v1,v2,l,0,cde,mxc)
           end if
           enddo  ! l loop
           endif
           enddo      ! v loop
           endif
           enddo           ! t loop
           enddo             ! n loop
```

Step 3p: **Read in values for extra supply projects.**

Note: nesp is the number of additional supply projects (frontiers type, or LNG).
supesy is the first year that the project is allowed.
supesn is the pointer to where the node is located.

```
           do 1850 e=1,nesp
           do 1840 t=1,ntme
           if(supesy(e).le.tme(t)) then
           n=supesn(e)
           if ((t.ge.ibegyr).and.(t.le.iendyr)) then
           call rget('ES##    ',val,valx,e,t,0,0,0,0,cde,mxc)
           else
           go to 1840
           end if

           do 1830 t1=t,ntme

           if ((t1.ge.ibegyr).and.(t1.le.iendyr)) then
           esvl(e,t1)=esvl(e,t1)+val
           endif

1830       continue
           endif
1840       continue
1850       continue


           do e=1,nesp
           do  t=1,ntme
           if(supesy(e).le.tme(t)) then
           n=supesn(e)
           if ((t.ge.ibegyr).and.(t.le.iendyr)) then
           write(*,*)'e,t,esvl(e,t)=',e,t,esvl(e,t)
           endif
           endif
           enddo
           enddo
```

Step 3q: **Read in values for operating level for peak supply.**

```
        do p=1,npks
        do r=1,ndrg
        n=dmnpnt(r)
        do t=1,ntme
        do l=1,nlds
        do k=1,2
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        call rget('PKO#####',pkovl(p,n,t,l,k),valx,
*       ,n,t,l,k,0,cde,mxc)
        end if
        enddo ! k loop
        enddo   ! l loop
        enddo     ! t loop
        enddo       ! r loop
        enddo         ! p loop
```

Step 3r: **Read in values for investment level for peak supply.**

Note: pklc is the levelized cost.
pkfc is the fixed O & M cost
npks is the number of peak supply options.
pkfyr is the first year available.

```
        do p=1,npks
        do r=1,ndrg
        n=dmnpnt(r)
        do t=1,ntme
        do k=1,2
        val = pklc(p,n,k)+pkfc(p,n,k)
        if (pkfyr(p,n,k) .le. tme(t)) then
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        call rget('PKI#### ',pkivl(p,n,t,k),valx,
*       ,n,t,k,0,0,cde,mxc)
        end if
        end if
        enddo    ! k loop
        enddo      ! t loop
        enddo        ! r loop
        enddo          ! p loop
```

Step 3s: **Create vectors that convert costs to present value.**

```
        do 2050 t=1,ntme
        do 2040 l=1,nlds
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        call rget('OD##    ',odvl(t,l),valx,t,l,0,0,0,0,cde,mxc)
        end if
2040    continue
        if ((t.ge.ibegyr).and.(t.le.iendyr)) then
        call rget('OA#     ',oavl(t),valx,t,0,0,0,0,0,cde,mxc)
        end if
2050    continue
```

Step 3t: **Read in supply values by region.**

```
        do 1995 t1=1,ntme
        do 1990 k=1,nsup
```

```
      ulmt=max(0.0,1.0-pscale*(nsps-suppas(k)))
      if(ulmt.gt.0.0) then
      if ((t1.ge.ibegyr).and.(t1.le.iendyr)) then
      call rget('SP##    ',spvl(k,t1),valx,k,t1,0,0,0,0,cde,mxc)
      end if
      spvl(k,t1)=spvl(k,t1)/vscls
      endif
1990  continue
1995  continue
```

Step 3u: Close 'gasall.prt' and return.

```
      close(14)
      return
      end
```

SUBROUTINE: INTRPS.FOR

CALLED BY: INTRPT.FOR (produces output files)

CALLS: PPRICE2.FOR (is used in the calculation of petroleum product prices)

READS: **‘**pflag.spc’ (is the flag for printing certain intermediate outputs)
‘ctgprc.dat’ (no longer used)
‘prd_prc.spc’ (contains revisions to the petroleum product prices)

CREATES: ‘gsamsln.fle’ (solution file derived from the Integrating Module)

MAIN THEME: This subroutine produces the summary supply and demand reports.

Step 1a: **Open ‘pflag.spc’. ‘pflag.spc’ is the flag for printing certain outputs (print = 1, do not print = 0). Read in data. Close file**

```
        open(unit=89,file='pflag.spc')
        rewind(89)
        read(89,1) pflag ! E&P print flag option, can overwrite
        read(89,1) pflag ! D&I print flag option
        format(I1)
        close(89)
```

Note: ldsdy is the number of days in gas load seasons.
nesp is the number of additional supply projects (frontiers type or LNG).
ndms is the number of demand sectors.
dmnctg is the citygate wholesale price for region n at time t.
lnkfus is the fuel use (%) for the transport capacity increments.
tnfvl is the transportation forward flow read from the LP.

Step 2a: **Initialize supply and demand arrays.**

```
        do 1000 t=ibegyr,iendyr
        suptbs(t)=0.0
        pksup(t)=0.0
        othspt(t)=0.0
        suptsp(t)=0.0
        suppsp(t)=0.0
        suptop(t)=0.0
        suptlp(t)=0.0
        suptlpus(t)=0.0
        do 250 r=1,ndrg
        dmnreg(r,t)=0.0
        dmnctg(r,t)=0.0
250     continue
        do 260 c=1,ndms
        dmnsec(c,t)=0.0
        dmnmsp(c,t)=0.0
        dmnsecus(c,t)=0.0
260     continue
        dmntot(t)=0.0
        dmntotus(t)=0.0
        dmnctt(t)=0.0
        trnfu(t)=0.0
        strfu(t)=0.0

        strfuus(t)=0.0
```

Step 2b: Calculate regional and national supply volumes and prices by year.

```
500     do 590 s=1,nsrg
        suprsp(s,t)=0.0
        n=suppnt(s)
        supreg(s,t)=0.0
        do 512 t1=ibegyr,t
        if(t1.eq.ibegyr) then
        vleft=1.0
        else
        vleft=1.0-vdr**(tme(t1)-tme(t1-1))
        endif
        vdo=vleft*vdr**(tme(t)-tme(t1))
        do 510 k=1,nsup

        if((t.eq.4).and.(s.eq.1)) then
        write(*,*) supreg(s,t),suptot(t,s,k),k,spvl(k,t1),t1,vdo

        endif

        supreg(s,t)=supreg(s,t)+suptot(t,s,k)*365./1000.*spvl(k,t1)*
*       vdo

        if((s.eq.14).and.(t.eq.4))then
        write(*,*)'k,supreg,suptot,spvl,vdo'
        write(*,*)nndnme(n),k,supreg(s,t),suptot(t,s,k),spvl(k,t1),vdo
        endif

        do 505 l=1,nlds
        suprsp(s,t)=suprsp(s,t)+(suptot(t,s,k)*supldf(s,l)*
*       spvl(k,t1))*vdo*
*       (ldsdy(l)/1000.0)*(mddl(n,t,l)-gthcst(s))
        if((s.eq.14).and.(t.eq.4))then

        write(*,*)'n,s,t,t1,k,l=',n,s,t,t1,k,l
        write(*,*)'suptot(t,s,k),supldf(s,l),spvl(k,t1),vdo='
        write(*,*)suptot(t,s,k),supldf(s,l),spvl(k,t1),vdo
        write(*,*)'ldsdy(l),mddl(n,t,l),gthcst(s)='
        write(*,*)ldsdy(l),mddl(n,t,l),gthcst(s)
        write(*,*)'suprsp(s,t)'
        write(*,*)suprsp(s,t)
        endif

        if (suprsp(s,t).lt.0.0) then
        end if
505     continue
510     continue
512     continue
```

```
         suplsp(s,t)=supreg(s,t)*(lsepln/100.0)
         suptlp(t)=suptlp(t)+suplsp(s,t)
         if(supreg(s,t).gt.0.0) then
         suprsp(s,t)=suprsp(s,t)/supreg(s,t)
         else
         suprsp(s,t)=0.0
         endif
         suptbs(t)=suptbs(t)+supreg(s,t)
         if(nndnme(n).ne.canada(4).and.nndnme(n).ne.canada(5).and.
*        nndnme(n).ne.mexico(2)) then
         suptotus(t)=suptotus(t)+supreg(s,t)
         suptlpus(t)=suptlpus(t)+supreg(s,t)*(lsepln/100.0)
         endif
         suptsp(t)=suptsp(t)+supreg(s,t)*suprsp(s,t)
590      continue
         if(suptbs(t).gt.0.0) then
         suptsp(t)=suptsp(t)/suptbs(t)
         else
         suptsp(t)=0.0
         endif
```

Step 2c: **Calculate peak supply.**

Note: suppsp(t) is the weighted average price of peaking supply at time t. pksup(t) is the total quantity of peaking supply at time t.

```
         do p=1,npks
         do r=1,ndrg
         n=dmnpnt(r)
         do l=1,nlds
         do k=1,2
         pksup(t)=pksup(t)+pkovl(p,n,t,l,k)*ldsdy(l)/1000.
         suppsp(t)=suppsp(t)+(pkovl(p,n,t,l,k)*ldsdy(l)/1000.)*
*        ((pklc(p,n,k)/365.0)+pkvc(p,n,k)+pkfc(p,n ,k))
         enddo    ! k loop
         enddo      ! l loop
         enddo        ! r loop
         enddo          ! p loop

         if(pksup(t).gt.0.0) then
         suppsp(t)=suppsp(t)/pksup(t)
         else
         suppsp(t)=0.0
         endif
```

Step 2d: **Calculate Other Supply.**

```
         do 750 e=1,nesp
         othsup(e,t)=0.0
         suposp(e,t)=0.0
         n=supesn(e)
         do 740 l=1,nlds
         val=esvl(e,t)*ldsdy(l)/1000.0
         othsup(e,t)=othsup(e,t)+val
         suposp(e,t)=suposp(e,t)+val*mddl(n,t,l)
740      continue
         othspt(t)=othspt(t)+othsup(e,t)
         if(othsup(e,t).gt.0.0) then
         suposp(e,t)=suposp(e,t)/othsup(e,t)
```

```
          else
          suposp(e,t)=0.0
          endif
          suptop(t)=suptop(t)+othsup(e,t)*suposp(e,t)
750       continue
          if(othspt(t).gt.0.0) then
          suptop(t)=suptop(t)/othspt(t)
          else
          suptop(t)=0.0
          endif
```

Step 2e: Calculate total supply.

```
          totsup(t)=othspt(t)+pksup(t)+suptbs(t)
```

Step 2f: Calculate regional and national demand volumes and prices by region.

```
          do 850 r=1,ndrg
          n=dmnpnt(r)
          do 840 c=1,ndms
          dmndet(r,c,t)=0.0
          dmnmdp(r,c,t)=0.0
          do 810 l=1,nlds
          if(c.eq.1) then
          valt=rsdmn(r,t,l)
          elseif(c.eq.2) then
          valt=cmdmn(r,t,l)
          elseif(c.eq.3) then
          valt=ifdmn(r,t,l)
          else
          valt=eugas(r,t,l)
          endif
          valt=valt*ldsdy(l)/1000.0
          dmndet(r,c,t)=dmndet(r,c,t)+valt
          dmnmdp(r,c,t)=dmnmdp(r,c,t)+valt*(mddl(n,t,l)+dmsdmr(c,r))
          dmnctg(r,t)=dmnctg(r,t)+valt*mddl(n,t,l)
810       continue
          if(dmndet(r,c,t).gt.0.0) then
          dmnmdp(r,c,t)=dmnmdp(r,c,t)/dmndet(r,c,t)
          else
          dmnmdp(r,c,t)=0.0
          endif
          dmnreg(r,t)=dmnreg(r,t)+dmndet(r,c,t)
          dmnsec(c,t)=dmnsec(c,t)+dmndet(r,c,t)
          if(nndnme(n).ne.canada(1).and.nndnme(n).ne.canada(2).and.
*         nndnme(n).ne.canada(3).and.nndnme(n).ne.mexico(1)) then
          dmnsecus(c,t)=dmnsecus(c,t)+dmndet(r,c,t)
          dmntotus(t)=dmntotus(t)+dmndet(r,c,t)
          endif
          dmnmsp(c,t)=dmnmsp(c,t)+dmndet(r,c,t)*dmnmdp(r,c,t)
          dmntot(t)=dmntot(t)+dmndet(r,c,t)
840       continue
          dmnctt(t)=dmnctt(t)+dmnctg(r,t)
          if(dmnreg(r,t).gt.0.0) then
          dmnctg(r,t)=dmnctg(r,t)/dmnreg(r,t)
          else
          dmnctg(r,t)=0.0
          endif
850       continue
```

```
          do 860 c=1,ndms
          if(dmnsec(c,t).gt.0.0) then
          dmnmsp(c,t)=dmnmsp(c,t)/dmnsec(c,t)
          else
          dmnmsp(c,t)=0.0
          endif
860       continue
          if(dmntot(t).gt.0.0) then
          dmnctt(t)=dmnctt(t)/dmntot(t)
          else
          dmnctt(t)=0.0
          endif
```

Step 2g: Calculate transportation fuel use.

```
          do 950 q=1,nlnk
          val=lnkfus(q)/100.0
          do 940 l=1,nlds
          trnfu(t)=trnfu(t)+tnfvl(q,t,l)*ldsdy(l)/1000.*val
          trnfuus(t)=trnfuus(t)+tnfvl(q,t,l)*ldsdy(l)/1000.*val*cnus(q)
940       continue
950       continue
```

Step 2h: Calculate storage fuel use.

Note: strfus is the fuel used for injection/extraction (%)
sivl is the storage injection
strfu is the fuel useage

```
          do n=1,nnde
          do tt=ibegyr,iendyr
          if(nsto(n,tt).gt.0) then
          do v=1,nsto(n,tt)
          val=strfus(n,tt,v)/100.0
          do l=1,nlds
          strfu(t)=strfu(t)+sivl(n,tt,v,l)*ldsdy(l)/1000.*val
          if (nndnme(n).ne.canada(1).and.
1         nndnme(n).ne.canada(2).and.
2         nndnme(n).ne.canada(3).and.
3         nndnme(n).ne.mexico(1)) then
          strfuus(t)=strfuus(t)+sivl(n,tt,v,l)*
1         ldsdy(l)/1000.*val
          end if
          enddo ! l  loop
          enddo   ! v  loop
          endif
          enddo       ! tt loop
          enddo        ! n  loop
```

Step 2i: Calculate net demand.

```
          dmnnet(t)=trnfu(t)+dmntot(t)+suptlp(t)+strfu(t)
          dmnnetus(t)=trnfuus(t)+dmntotus(t)+suptlpus(t)+strfuus(t)
          netimp(t)=totsup(t)-(dmnnet(t)-dmnnetus(t))-suptotus(t)
          canimp(t)=netimp(t)-othspt(t)-pksup(t)
*         +othsup(1,t)+othsup(2,t)
```

Note: refmar is refinery margins.
regmar is regional margins.

Step 3a: Call PPRICE2.FOR to calculate industrial sector petroleum product prices.

```
        do 995 r=1,ndrg
        call pprice2(prccrd(t),refmar(1,t),regmar(1,2,r,t),
*       regmar(1,1,r,t),pprind(1,t,r))

        call pprice2(prccrd(t),refmar(2,t),regmar(2,2,r,t),
*       regmar(2,1,r,t),pprind(2,t,r))

        call pprice2(prccrd(t),refmar(3,t),regmar(3,2,r,t),
*       regmar(3,1,r,t),pprind(3,t,r))
```

Step 3b: Call PPRICE2.FOR to calculate electrical power generation petroleum product prices.

```
        call pprice2(prccrd(t),refmar(1,t),regmar(1,2,r,t),
*       0.0,pprele(1,t,r))

        call pprice2(prccrd(t),refmar(2,t),regmar(2,2,r,t),
*       0.0,pprele(2,t,r))

        call pprice2(prccrd(t),refmar(3,t),regmar(3,2,r,t),
*       0.0,pprele(3,t,r))

995     continue

1000    continue
```

Step 3c: Write out supply volumes by region and project.

```
        if(pflag.eq.1) write(22,1001) (tme(t),t=ibegyr,iendyr)
1001    format(' Supply Summary',t42,33(2x,i4,2x))
        if(pflag.eq.1) write(22,1002)
1002    format(' Supply Model')
        do 1100 s=1,nsrg
        n=suppnt(s)
        if(pflag.eq.1) write(22,1003) nndnme(n),(supreg(s,t),
*       t=ibegyr,iendyr)
1003    format(' Region: ',a20,t42,33(1x,f7.1))
1100    continue

        if(pflag.eq.1) write(22,1099) (suptotus(t),t=ibegyr,iendyr)
1099    format(' Total U.S. Supply Model',t42,33(1x,f7.1))

        if(pflag.eq.1) write(22,1101) (suptbs(t),t=ibegyr,iendyr)
1101    format(' Total Supply Model',t42,33(1x,f7.1))

        if(pflag.eq.1) write(22,1102)  (tme(t),t=ibegyr,iendyr)
1102    format(/' Supply Projects:',t42,33(2x,i4,2x))
        do 1200 e=1,nesp
        if(pflag.eq.1) write(22,1103) supenm(e),(othsup(e,t),
*       t=ibegyr,iendyr)
```

```
1103      format(' Project: ',a20,t42,33(1x,f7.1))
1200      continue
          if(pflag.eq.1) write(22,1201) (othspt(t),t=ibegyr,iendyr)
1201      format(' Total Supply Projects: ',t42,33(1x,f7.1))

          if(pflag.eq.1) write(22,1202) (pksup(t),t=ibegyr,iendyr)
1202      format(/' Peak Supplies: ',t42,33(1x,f7.1))
          if(pflag.eq.1) write(22,1203) (totsup(t),t=ibegyr,iendyr)
1203      format(/' Total Supplies: ',t42,33(1x,f7.1))
```

Step 3d: Write out demand volumes by region and sector.

```
          if(pflag.eq.1) write(22,1204)
1204      format(' --------------------------'/' '/
*          ' Demand by Region and Sector')
          do 1300 r=1,ndrg
          n=dmnpnt(r)
          if(pflag.eq.1) write(22,1205) nndnme(n),(tme(t),t=ibegyr,iendyr)
1205      format(/' Demand Region: ',a20,t42,33(2x,i4,2x))
          do 1250 c=1,ndms
          if(pflag.eq.1) write(22,1206) dmsnme(c),(dmndet(r,c,t),
*         t=ibegyr,iendyr)
1206      format(' Sector: ',a20,t42,33(1x,f7.1))
1250      continue
          if(pflag.eq.1) write(22,1251) (dmnreg(r,t),t=ibegyr,iendyr)
1251      format(' Total: ',t42,33(1x,f7.1))
```

Step 3e: Write out total demand volumes by sector.

```
1300      continue
          if(pflag.eq.1) write(22,1301) (tme(t),t=ibegyr,iendyr)
1301      format(/' Total for All Regions: ',t42,33(2x,i4,2x))
          do 1350 c=1,ndms
          if(pflag.eq.1) write(22,1306) dmsnme(c),(dmnsec(c,t),
*         t=ibegyr,iendyr)
1306      format(' Sector: ',a20,t42,33(1x,f7.1))
1350      continue
          if(pflag.eq.1) write(22,1351) (dmntot(t),t=ibegyr,iendyr)
1351      format(' Total: ',t42,33(1x,f7.1))

          if(pflag.eq.1) write(22,1352) (trnfu(t),t=ibegyr,iendyr)
1352      format(/' Transport Fuel Use: ',t42,33(1x,f7.1))
          if(pflag.eq.1) write(22,1353) (strfu(t),t=ibegyr,iendyr)
1353      format(' Storage Fuel Use: ',t42,33(1x,f7.1))
          if(pflag.eq.1) write(22,1355) (suptlp(t),t=ibegyr,iendyr)
1355      format(' Lease & Plant: ',t42,33(1x,f7.1))
          if(pflag.eq.1) write(22,1354) (dmnnet(t),t=ibegyr,iendyr)
1354      format(/' Net Demand: ',t42,33(1x,f7.1))
```

Step 3f: Write out US demand volumes by sector.

```
          if(pflag.eq.1) write(22,1361) (tme(t),t=ibegyr,iendyr)
1361      format(/' Total for United States: ',t42,33(2x,i4,2x))
          do 1360 c=1,ndms
          if(pflag.eq.1) write(22,1306) dmsnme(c),(dmnsecus(c,t),
*         t=ibegyr,iendyr)
1360      continue
          if(pflag.eq.1) write(22,1351) (dmntotus(t),t=ibegyr,iendyr)
          if(pflag.eq.1) write(22,1352) (trnfuus(t),t=ibegyr,iendyr)
          if(pflag.eq.1) write(22,1353) (strfuus(t),t=ibegyr,iendyr)
```

```
        if(pflag.eq.1) write(22,1355) (suptlpus(t),t=ibegyr,iendyr)
        if(pflag.eq.1) write(22,1354) (dmnnetus(t),t=ibegyr,iendyr)
        if(pflag.eq.1) write(22,1366) (netimp(t),t=ibegyr,iendyr)
        if(pflag.eq.1) write(22,1367) (canimp(t),t=ibegyr,iendyr)
1366    format(/' Total Imports: ',t42,33(1x,f7.1))
1367    format(/' Imports from Canada: ',t42,33(1x,f7.1))
```

Step 3g: Write out supply prices by region and project.

```
        if(pflag.eq.1) write(22,1401) (tme(t),t=ibegyr,iendyr)
1401    format(/' ----------------------------'/' '/
*       ' Supply Price Summary',t42,33(2x,i4,2x))
        if(pflag.eq.1) write(22,1002)
        do 1500 s=1,nsrg
        n=suppnt(s)
        if(pflag.eq.1) write(22,1403) nndnme(n),(suprsp(s,t),
        * t=ibegyr,iendyr)
1403    format(' Region: ',a20,t42,33(1x,f7.2))
1500    continue
        if(pflag.eq.1) write(22,1501) (suptsp(t),t=ibegyr,iendyr)
1501    format(' Average Supply Model',t42,33(1x,f7.2))
        if(pflag.eq.1) write(22,1502) (tme(t),t=ibegyr,iendyr)
1502    format(/' Supply Projects:',t42,33(2x,i4,2x))
        do 1600 e=1,nesp
        if(pflag.eq.1) write(22,1503) supenm(e),(suposp(e,t),
*       t=ibegyr,iendyr)
1503    format(' Project: ',a20,t42,33(1x,f7.2))
1600    continue
        if(pflag.eq.1) write(22,1601) (suptop(t),t=ibegyr,iendyr)
1601    format(' Average Supply Projects: ',t42,33(1x,f7.2))

        if(pflag.eq.1) write(22,1602) (suppsp(t),t=ibegyr,iendyr)
1602    format(/' Peak Supplies : ',t42,33(1x,f7.2))
        if(pflag.eq.1) write(22,1603)
1603    format(/' ---------------------------------'/' ')
```

Step 3h: Write out demand prices (natural gas prices, industrial petroleum product prices, and electric power generation petroleum product prices) by region and sector.

```
        if(pflag.eq.1) write(22,1604)
1604    format(' Regional Wholesale Hub and End-Use Prices by Region'
*       ' and Sector')
        do 1700 r=1,ndrg
        n=dmnpnt(r)
        if(pflag.eq.1) write(22,1605) nndnme(n),(tme(t),t=ibegyr,iendyr)
1605    format(/' Demand Region: ',a20,t42,33(2x,i4,2x))
        do 1650 c=1,ndms
        if(pflag.eq.1) write(22,1606) dmsnme(c),(dmnmdp(r,c,t),
*       t=ibegyr,iendyr)
1606    format(' Sector: ',a20,t42,33(1x,f7.2))
1650    continue

        if(pflag.eq.1) write(22,1651) (dmnctg(r,t),t=ibegyr,iendyr)
1651    format(' Wholesale Price: ',t42,33(1x,f7.2))

        if(pflag.eq.1) write(22,1652)
1652    format(/' Industrial Petroleum Product Prices: ')
        if(pflag.eq.1) write(22,1653) (pprind(1,t,r),t=ibegyr,iendyr)
1653    format('    Distillate: ',t42,33(1x,f7.2))
```

```
        if(pflag.eq.1) write(22,1654) (pprind(2,t,r),t=ibegyr,iendyr)
1654    format('    Low Sulfur Resid: ',t42,33(1x,f7.2))
        if(pflag.eq.1) write(22,1655) (pprind(3,t,r),t=ibegyr,iendyr)
1655    format('    High Sulfur Resid: ',t42,33(1x,f7.2))

        if(pflag.eq.1) write(22,1656)
1656    format(/' Electrical Power Gen. Petroleum Product Prices: ')
        if(pflag.eq.1) write(22,1657) (pprele(1,t,r),t=ibegyr,iendyr)
1657    format('    Distillate: ',t42,33(1x,f7.2))
        if(pflag.eq.1) write(22,1658) (pprele(2,t,r),t=ibegyr,iendyr)
1658    format('    Low Sulfur Resid: ',t42,33(1x,f7.2))
        if(pflag.eq.1) write(22,1659) (pprele(3,t,r),t=ibegyr,iendyr)
1659    format('    High Sulfur Resid: ',t42,33(1x,f7.2))

1700    continue
        if(pflag.eq.1) write(22,1701) (tme(t),t=ibegyr,iendyr)
1701    format(/' Total for All Regions: ',t42,33(2x,i4,2x))
        do 1750 c=1,ndms
        if(pflag.eq.1) write(22,1706) dmsnme(c),(dmnmsp(c,t),
*       t=ibegyr,iendyr)
1706    format(' Sector: ',a20,t42,33(1x,f7.2))
1750    continue
        if(pflag.eq.1) write(22,1751) (dmnctt(t),t=ibegyr,iendyr)
1751    format(' Average Wholesale: ',t42,33(1x,f7.2))
```

Step 3i: Write detailed electric utility reports – capacity and utilization by electricity load period, year, and region.

```
2000    do 2100 t=ibegyr,iendyr
        do 2080 f=1,7
        do 2070 k=1,2
        eucapt(t,f,k)=0.0
        do 2060 z=1,4
        euutot(t,f,z)=0.0
        euutgt(t,f,z)=0.0
2060    continue
2070    continue
2080    continue
2100    continue
        do 2200 r=1,ndrg+1
        if(r.le.ndrg) then
        n=dmnpnt(r)
        if(pflag.eq.1) write(22,2101) nndnme(n)
2101    format(/' -------------------------------------------------'/
*       /' '/' Region: ',a20,' Electric Generation Model Results')
        else
        if(pflag.eq.1) write(22,2102)
2102    format(' -------------------------------------------------'/' '/
*       ' Region: Total',15x,' Electric Generation Model Results')
        endif
        do 2106 t=ibegyr,iendyr
        eucapx(t,1)=0.0
        eucapx(t,2)=0.0
        eucpxo(t)=0.0
        eucpxg(t)=0.0
2106    continue
        do 2180 f=1,7
        if(pflag.eq.1) write(22,2103) eutype(f),(tme(t),t=ibegyr,iendyr)
2103    format(/' Generation Type: ',a12,t42,33(2x,i4,2x))
        if(r.le.ndrg) then
        if(pflag.eq.1) write(22,2104) (eucap(r,t,f,1),t=ibegyr,iendyr)
```

```
2104    format(' Existing Capacity: (gigawatts) ',t42,33(1x,f7.2))
        if(pflag.eq.1) write(22,2105) (eucap(r,t,f,2),t=ibegyr,iendyr)
2105    format(' New Capacity: (gigawatts) ',t42,33(1x,f7.2))
        do 2110 t=ibegyr,iendyr
        eucapt(t,f,1)=eucapt(t,f,1)+eucap(r,t,f,1)
        eucapt(t,f,2)=eucapt(t,f,2)+eucap(r,t,f,2)
        eucptl(t)=eucap(r,t,f,1)+eucap(r,t,f,2)
        eucapx(t,1)=eucapx(t,1)+eucap(r,t,f,1)
        eucapx(t,2)=eucapx(t,2)+eucap(r,t,f,2)
2110    continue
        else
        if(pflag.eq.1) write(22,2104) (eucapt(t,f,1),t=ibegyr,iendyr)
        if(pflag.eq.1) write(22,2105) (eucapt(t,f,2),t=ibegyr,iendyr)
        do 2120 t=ibegyr,iendyr
        eucptl(t)=eucapt(t,f,1)+eucapt(t,f,2)
        eucapx(t,1)=eucapx(t,1)+eucapt(t,f,1)
        eucapx(t,2)=eucapx(t,2)+eucapt(t,f,2)
2120    continue
        endif
        if(pflag.eq.1) write(22,2121) (eucptl(t),t=ibegyr,iendyr)
2121    format(' Total Capacity: ',t42,33(1x,f7.2))

        if(pflag.eq.1) write(22,2122)
2122    format(' Capacity Utilization')
        do 2150 z=1,4
        do 2130 t=ibegyr,iendyr
        if(r.le.ndrg) then
        eucpxo(t)=eucpxo(t)+euutlo(r,t,f,z)
        eucpxg(t)=eucpxg(t)+euutlg(r,t,f,z)
        if(eucptl(t).gt.0.0) then
        val=eucptl(t)*ndeu(z)/365.*8760.0/1000.0
        eucpto(t)=euutlo(r,t,f,z)/val*100.0
        eucptg(t)=euutlg(r,t,f,z)/val*100.0
        else
        eucpto(t)=0.0
        eucptg(t)=0.0
        endif
        euutot(t,f,z)=euutot(t,f,z)+euutlo(r,t,f,z)
        euutgt(t,f,z)=euutgt(t,f,z)+euutlg(r,t,f,z)
        else
        eucpxo(t)=eucpxo(t)+euutot(t,f,z)
        eucpxg(t)=eucpxg(t)+euutgt(t,f,z)
        if(eucptl(t).gt.0.0) then
        val=eucptl(t)*ndeu(z)/365.*8760.0/1000.0
        eucpto(t)=euutot(t,f,z)/val*100.0
        eucptg(t)=euutgt(t,f,z)/val*100.0
        else
        eucpto(t)=0.0
        eucptg(t)=0.0
        endif
        endif
2130    continue
        if(pflag.eq.1) write(22,2123) perd(z),(eucptg(t),
*       t=ibegyr,iendyr)
2123    format(' Period: ',a12,' Gas: ',t42,33(1x,f7.1))
        if(pflag.eq.1) write(22,2124) (eucpto(t),t=ibegyr,iendyr)
2124    format('         ',12x,' Other: ',t42,33(1x,f7.1))
2150    continue
2180    continue
        if(pflag.eq.1) write(22,2181) (tme(t),t=ibegyr,iendyr)
2181    format(/' Total Generation:',t42,33(2x,i4,2x))
        if(pflag.eq.1) write(22,2104) (eucapx(t,1),t=ibegyr,iendyr)
        if(pflag.eq.1) write(22,2105) (eucapx(t,2),t=ibegyr,iendyr)
        do 2185 t=ibegyr,iendyr
        eucptl(t)=eucapx(t,1)+eucapx(t,2)
2185    continue
        if(pflag.eq.1) write(22,2121) (eucptl(t),t=ibegyr,iendyr)
        if(pflag.eq.1) write(22,2186) (eucpxo(t),t=ibegyr,iendyr)
2186    format(' '/' Non-Gas Generation (billion kwhs)',t42,33(1x,f7.1))
        if(pflag.eq.1) write(22,2187) (eucpxg(t),t=ibegyr,iendyr)
2187    format(' '/' Gas Generation (billion kwhs)',t42,33(1x,f7.1))
        do 2190 t=ibegyr,iendyr
        if(eucptl(t).gt.0.0) then
        eucptl(t)=(eucpxo(t)+eucpxg(t))/(eucptl(t)*8760/1000.0)*100.0
        else
        eucptl(t)=0.0
        endif
2190    continue
        if(pflag.eq.1) write(22,2191) (eucptl(t),t=ibegyr,iendyr)
2191    format(' Overall Capacity Factor(%)',t42,33(1x,f7.1))
2200    continue
```

SUBROUTINE: INTRPT.FOR

CALLED BY:	None
CALLS:	ERRMSG.FOR (prints out error and warning messages) INTRDT.FOR (reads in the transportation and other non-demand data) INTRPD.FOR (writes out detailed transportation reports) INTRPG.FOR (reads results from the LP solution file) INTRPS.FOR (produces the summary supply and demand reports) INTRSA.FOR (writes out storage reservoir level activities by season and year) INTRSC.FOR (produces the storage costs report) SUP_REP.FOR (produce reports for peak supply, propane/air and LNG, storage, and pipeline usage)
READS:	None
CREATES:	'gsamsln.rpt' (contains a summary of flows and node by node material balance). 'gsamsln.fle' (contains a summary of demand, supply and prices)
MAIN THEME:	This routine produces the reports generated from the demand integrating module.

Step 1a: **Initialize error message routine and read in input specifications.**

```
call errmsg(0,0)
call intrdt
```

Step 2a: **Read in solution from LP solution file.**

```
call intrpg
```

Step 3a: **Open report files.**

```
open(21,file='gsamsln.rpt')
open(22,file='gsamsln.fle')
```

Step 4a: **Produce detailed transportation reports.**

```
call intrpd
```

Step 5a: Produce the summary annual supply and demand reports.

```
call intrps
```

Step 6a: Produce the storage activity summary report.

```
call intrsa
```

Step 7a: Produce the storage costs summary report.

```
call intrsc
```

Step 8a: Produce the report that compares peak supply (propane/air and LNG), storage, and pipeline usages.

```
call sup_rep
stop
end
```

SUBROUTINE: INTRSA.FOR

CALLED BY: INTRPT.FOR (produces output files)

CALLS: FINDFYR.FOR (searches for the first year the storage reservoir is active)

READS: None

CREATES: **‘**gsamsln.sta’ (contains a summary of storage activity)

MAIN THEME: Writes out storage reservoir level activities by season and year.

Note: ndrg is the number of demand regions.
nndnme is an identifier of nodes.
nlds is the number of load segments.
mxnres is the maxinum number of storage reservoirs for a fixed node, year.
nsto is the number of reservoirs for year t at node n.
ndin is the number of demand increments.
strind is the identifier of storage.

Step 1a: **Open ‘gsamsln.sta’ and write headers.**

```
        open(unit=88,file='gsamsln.sta')
        tol = 0.0

        write(88,500)
        write(88,501)
        write(88,500)
        write(88,502)
        write(88,503)decline*100.0,storuse*100.0

500     format(/' ==============================================')
501     format(/' Storage Activity Summary Report')
502     format(' ')
503     format(' Decline factor: ',1x,f5.1,'%'1x,
*       ' Storage Usage: ',f5.1,'%')
```

Note: Reports by Demand Region for Storage Extraction/Injection Activities.

```
        do 1500 n=1,ndrg
        write(88,600)
600     format(/' --------------------------------------------')
        write(88,801) nndnme(n)
801     format(' Demand Region : ',a20)

        do 1490 l=1,nlds+1
        if(l.le.nlds) then
        write(88,802) l,ldsdy(l)
802     format(/' Load Period: ',i2,' Number of Days:',i3,
1       ' (mmcf/day)')
        else
        write(88,803)
803     format(/' Annual Averages (Bcf) Note: Storage Injection',/,
*       'Values Are **Not** Less Fuel Usage',/)
        endif

        write(88,804) (tme(t),t=ibegyr,iendyr)
804     format(' ',t42,33(3x,i4,2x))
```

Step 1b: **Initialize arrays.**

```
        do t=ibegyr,iendyr
        ndsex(t)    = 0.0
        ndsin(t)    = 0.0
        do v=1,mxnres
        ndin(v,t) = 0.0
        end do  ! v loop
        end do  ! t loop
```

Step 1c: **Calculate number of storage reservoirs for each (node, time) combination.**

Note: l is the gas load segment index (l=1,nlds)=> MMCF/day data (l=nlds+1)=> BCF/year data

sevl is storage extraction value, read from LP (MMCF/day). ndsex is storage extraction summed over t (years).

```
        nres = 0
        do t=ibegyr,iendyr
        if (nsto(n,t).gt.nres) nres = nsto(n,t)
        enddo ! t loop
```

Step 1d: **Write out storage extraction.**

```
        write(88,862)
        write(88,863)
        write(88,864)
862     format(' Storage Extraction')
863     format(t21,' First',' Available ')
864     format(t21,' Year ',' Capacity  ', t42,
*       '  Storage Usage')

        do t=ibegyr,iendyr

        if(nsto(n,t).gt.0) then
        do v = 1,nsto(n,t)
        if(l.le.nlds) then        ! in MMCF/day
        ndin(v,t)=sevl(n,t,v,l)
        vcp(v,t) =strvcp(n,t,v)*strecp(n,t,v,l)/100.0
        else
        ndin(v,t)= 0.0            ! in BCF/year
        vcp(v,t) = 0.0
        sumv     = 0.0
        do l1=1,nlds
        sumv = sumv+ sevl(n,t,v,l1)
        enddo

        do  l1=1,nlds
        ndin(v,t)=ndin(v,t)+sevl(n,t,v,l1)*ldsdy(l1)
*       /1000.0
        if (sumv.gt.tol) then
        vcp(v,t) =vcp(v,t)+
*       (strvcp(n,t,v)*(strecp(n,t,v,l1)/100.0)*
*       ldsdy(l1))/1000.
        endif
        enddo ! l1 loop
```

Note: Can not exceed the volume capacity and can't exceed the annual (weighted) daily extraction rate.

```
        vcp(v,t) = min(vcp(v,t),strvcp(n,t,v)/1000.)
        endif
        ndsex(t)=ndsex(t)+ndin(v,t)
        enddo    ! v loop
        endif
        enddo  ! t loop

        do v=1,nres

        call findfyr(ffyr)

        fyr=max(ibegyr,ffyr)

        write(88,875)strind(n,fyr,v),
*       tme(fyr),
*        vcp(v,fyr),(ndin(v,t),
*       t=ibegyr,iendyr)

875     format (' ',a20,1x,i4,1x,f8.2,t42,33(1x,f8.2))
        end do  ! v loop
        write(88,881) (ndsex(t),t=ibegyr,iendyr)
881     format(' Total Storage Extraction: ',t42,33(1x,f8.2))
```

Step 1e: Write out storage injection.

Note: l is the gas load segment index (l=1,nlds)=> MMCF/day data
(l=nlds+1)=> BCF/year data

sivl is the storage injection value, read from LP (MMCF/day).
ndsin is the storage injection summed over t (years).

nres as calculated in the storage extraction section should
be the same as calculated below for every reservoir:
storage extraction level >0 <==> storage injection level

```
        write(88,1022)
1022    format(/' Storage Injection')
```

```
          do t=ibegyr,iendyr
          if(nsto(n,t).gt.0) then
          do v = 1,nsto(n,t)
          if(l.le.nlds) then        ! in MMCF/day
          ndin(v,t)=sivl(n,t,v,l)
          vcp(v,t) = strvcp(n,t,v)*stricp(n,t,v,l)/100.0
          else
          ndin(v,t)=0.0              ! in BCF/year
          vcp(v,t) =0.0
          do  l1=1,nlds
          ndin(v,t)=ndin(v,t)+sivl(n,t,v,l1)*ldsdy(l1)
*         /1000.0
          vcp(v,t) =vcp(v,t)+
*         (strvcp(n,t,v)*(stricp(n,t,v,l1)/100.0)*
*         ldsdy(l1))/1000.0
          enddo ! l1 loop
```

Note: Can not exceed the volume capacity and can not exceed the annual (weighted) daily injection rate.

```
          vcp(v,t) = min(vcp(v,t),strvcp(n,t,v)/1000.)
          endif

          ndsin(t)=ndsin(t)+ndin(v,t)
          enddo    ! v loop
          endif
          enddo        ! t loop

          do v=1,nres

          call findfyr(ffyr)

          fyr=max(ibegyr,ffyr)

          write(88,875)strind(n,fyr,v),
*         tme(fyr),
*         vcp(v,fyr),(ndin(v,t),
*         t=ibegyr,iendyr)
          end do  ! v loop
          write(88,1041) (ndsin(t),t=ibegyr,iendyr)
1041      format(' Total Storage Injection: ',t42,33(1x,f8.2))


1490      continue         ! load loop l
1500      continue         ! node loop n
```

Step 1f: **Close ‘gsamsln.sta’.**

```
          close(88)
          return
          end
```

SUBROUTINE: INTRSC.FOR

CALLED BY: INTRPT.FOR (produces output files)

CALLS: None

READS: None

CREATES: ‘gsamsln.stc’ (contains a summary of the storage cost)

MAIN THEME: This subroutine produces the Storage Costs Report.

Note: ndrg is the number of demand regions.
ntme is the number of time periods.
nsto is the number of reservoirs for year t at node n.
fuscst is the fuel usage/shrinkage cost in $.
storext is storage extraction activity in MCF.
totext is the total storage extraction for the year in MCF.
strfus is the fuel used for extraction/injection.
strind is an identifier of storage.

Step 1a: **Open ‘gsamsln.stc’ and write headers.**

```
        open(unit=88,file='gsamsln.stc')

        write(88,500)
        write(88,501)
        write(88,500)
        write(88,502)
        write(88,503)decline*100.0,storuse*100.0

500     format(/' ==============================================')
501     format(/' Storage Costs Summary Report')
502     format(' ')
503     format(' Decline factor: ',1x,f5.1,'%'1x,
*       ' Storage Usage: ',f5.1,'%')

        do n=1,ndrg

        write(88,600)
600     format(/' --------------------------------------------')
        write(88,801) nndnme(n)
801     format(' Demand Region : ',a20)

        write(88,803)
803     format(/' Annual Values ($/MCF)')

        write(88,804) (tme(t),t=ibegyr,iendyr)
804     format(' ',t43,33(3x,i4,2x))
```

Note: sevl is the storage extraction value, read from LP.

storext is the storage extraction activity.
totext is the total storage extraction for the year.
strcst is the levelized investment cost.
strfom is the fixed O&M cost.
strvom is the variable O&M.
fuscst is the fuel useage/shrinkage cost.
nres is the maximum number of reservoirs for node n for all times t.

Step 1b: **Calculate the number of storage reservoirs for each (node, time) combination.**

```
nres = 0
do t=1,ntme
if (nsto(n,t).gt.nres) nres = nsto(n,t)
enddo ! t loop
```

Step 1c: **Initialize unit costs at "numerical infinity".**

```
do t=1,ntme
do v=1,nres
unitcst(v,t) = 9999.99
end do  ! v loop
end do  ! t loop
```

Step 1d: **Initialize values.**

```
do t=1,ntme
if(nsto(n,t).gt.0) then
do v = 1,nsto(n,t)
levcst  = 0.0
fomcst  = 0.0
vomcst  = 0.0
fuscst  = 0.0
storext = 0.0
totext  = 0.0
```

Step 1e: **Compute unit costs and write out values.**

```
do  l=1,nlds
storext = sevl(n,t,v,l)*ldsdy(l)*1000.0
totext  = totext + storext
enddo ! l loop
```

Note: Use off-season (season 4) price for fuel costs

```
        fuscst=duals(n,t,4)*totext*strfus(n,t,v)/100.0

        if (totext.gt.1.d-3) then
        unitcst(v,t)=strcst(n,t,v)+
*       strfom(n,t,v)+strvom(n,t,v)+(fuscst)/totext
        else
        unitcst(v,t)=9999.99
        end if
```

```
          enddo    ! v loop
          endif
          enddo         ! t loop

          do v=1,nres
          if (unitcst(v,1) .eq. 0.0) then
          write (*,*)'n,t,v,unitcst=',n,t,v,unitcst(v,t)
          end if
          write(88,875)strind(n,ntme,v),(unitcst(v,t),t=ibegyr,iendyr)
875       format (' ',a20,t42,33(1x,f8.2))
          end do  ! v loop
          enddo          ! n loop
```

Step 1f: Close ‘gsamsln.stc’.

```
          close(88)
          return
          end
```

SUBROUTINE: INTRVS.FOR

CALLED BY: None

CALLS: ERRMSG.FOR (prints out error and warning messages)
GASHIST.FOR (contains the gas historical price routine and reads in gas prices, by track, from 1993 to the beginning year of the GSAM Integrating Model)
INTRDT.FOR (reads in the transportation and other non-demand data for the LP)
INTRPG.FOR (reads results from the solution file)

READS: 'oduals.spc' (contains the old dual prices for each region)

CREATES: **'**gasprc.new' (contains the gas prices derived from the LP run to be used in the Exploration and Production module)
'oduals.spc' (contains the old dual prices for each region)

MAIN THEME: Checks for convergence of overall GSAM iterative procedure for computing market equilibrium values.

Note: nsrg is the number of supply regions.
suppnt is the pointer to the node where the supply region is located.
nlds is the number of load segments.
gthcst is the gathering cost.
ldsdy is the number of days in a gas load season.
supnpr is the input supply price.
nsps is the idex of pass through the model with new supply.
hisprc is the historical prices.

Step 1a: **Initialize error message routine. ERRMSG.FOR prints out error and warning messages. (see ERRMSG.FOR)**

```
call errmsg(0,0)
```

Step 2a: **Read in input specifications.**

```
call intrdt
```

Step 3a: **Read in solution from LP solution file. INTRPG.FOR reads results from the LP solution file.**

```
call intrpg
```

Step 3b: Initialize historical dual prices.

```
        do t1=1,ibegyr-1
        histdual(t1)=0.0
        enddo
```

Step 4a: Open 'oduals.spc'. 'oduals.spc' contains the old dual prices for each region and time period from the previous run. Read in the old dual prices from the previous pass. Close file.

```
        open (unit=98,file='oduals.spc')
        do s=1,nsrg
        n=suppnt(s)
400     read(98,405) nname,(lastdual(t1,s),t1=1,iendyr)
405     format(a20,33(1x,f7.3))
        if(nname.ne.nndnme(n)) then
        write(*,430) nname
        call errmsg(3,204)
430     format(' Demand region: ',a20)
        endif
        enddo
        close(98)
```

Step 4b: Set up supply price vectors for new solution - first average price of supply in LP solution.

```
        valmxa=0.0
500     do 550 s=1,nsrg
        do 545 t1=ibegyr,iendyr
        if(t1.eq.ibegyr) then
        vleft=1.0
        else
        vleft=1.0-vdr**(tme(t1)-tme(t1-1))
        endif
        do 540 t=t1,iendyr
        vdo=vleft*vdr**(tme(t)-tme(t1))
        if(t1.eq.1) supnpr(t,s,2)=0.0  ! new code 08-01-96
        do 530 k=1,nsup
        supnpr(t,s,2)=supnpr(t,s,2)+spvl(k,t1)*supprc(t,s,k)*vdo !new code
        valmxa=amax1(valmxa,spvl(k,t1))
530     continue
540     continue
545     continue
550     continue
```

Step 4c: Second, get marginal price of demand for gas at the supply nodes in the LP solution.

```
        do 650 s=1,nsrg
        n=suppnt(s)
        do 640 t=ibegyr,iendyr
        supnpr(t,s,3)=0.0  ! new code
        do 630 l=1,nlds
        if (mddl(n,t,l)-gthcst(s) .le. 0.0) then
        mddl(n,t,l) = 1.00 + gthcst(s)
        endif
        supnpr(t,s,3)=supnpr(t,s,3)+
*       (mddl(n,t,l)-gthcst(s))*supldf(s,l)*ldsdy(l)/365.

630     continue
640     continue
650     continue
```

Step 4d: **Set up intermediate supply price vectors for new solution and estimate maximum change in last pass.**

```
        valmxb=0.0
        do 750 s=1,nsrg
        do 740 t=ibegyr,iendyr
```

Note: Force the first price to be $0.10 as well as ascending order for other prices. Save the 2nd price in a different array. sup_save is the supply price to be written as the 5th price track is gasprc.new file.

```
        vala=amax1(supnpr(t,s,2),0.2)
        valb=amax1(supnpr(t,s,3),0.2)

        n=suppnt(s)
        if((nndnme(n).eq.'Alberta').or.
*       (nndnme(n).eq.'British Columbia')) then
        if(lastdual(t,s).ne.0.00)
*       valb=(1.0*valb+2.0*lastdual(t,s))/3.0
        else
        if(lastdual(t,s).ne.0.00)
*       valb=(1.0*valb+1.0*lastdual(t,s))/2.0
        end if

        lastdual(t,s)=amax1(supnpr(t,s,3),0.2)

        sup_save(t,s,2) = supnpr(t,s,2)

        val=abs(vala-valb)

        supnpr(t,s,2) = valb - fac
        supnpr(t,s,3) = valb + fac
        supnpr(t,s,1) = amin1(0.50*supnpr(t,s,2),1.00)
        supnpr(t,s,4) = amin1(supnpr(t,s,3)*1.50,3.00)

        supnpr(t,s,5)=valb


        if(val.lt.0.0) val=-val
        valmxb=amax1(valmxb,val)
740     continue
750     continue
```

Step 4e: **Open ‘oduals.spc’. Write out dual prices from this pass to be used for the next pass. Close file.**

```
        open (unit=98,file='oduals.spc')
        do s=1,nsrg
        n=suppnt(s)
770     write(98,775) nndnme(n),
*       (histdual(t1),t1=1,ibegyr-1),
*       (lastdual(t1,s),t1=ibegyr,iendyr)
775     format(a20,33(1x,f7.3))
        enddo
        close(98)
```

Note: Output New Supply Price Vectors if Solution Tolerances Not Within Minimum Allowed.

Step 5a: **Call GASHIST.FOR. This subroutine reads in gas prices by region, track, and year for years prior to GSAM model years.**

```
        call gashist
```

Step 6a: **Write out gas prices to gasprc.new and end ro utine.**

```
        if (valmxb.gt.0.07) then
        open(31,file='gasprc.new')

        nsps=nsps+1
        do 850 ii=1,4
        do 840 s=1,nsrg
        n=suppnt(s)

        if (nndnme(n).eq.'North Alaska        ') then
        do t=ibegyr,iendyr
        supnpr(t,s,ii) = 0.10
        enddo
        endif

        write(31,801) nndnme(n),nsps,ii,
*       (hisprc(t,s,ii),t=1,(ibegyr-1)),
*       (supnpr(t,s,ii),t=ibegyr,iendyr)

801     format(a20,2i3,33(1x,f7.3))
840     continue
850     continue



        do 851 ii=2,2
        do 841 s=1,nsrg
        n=suppnt(s)
```

Note: Change for North Alaska, prices should be 0.10.

```
        if (nndnme(n).eq.'North Alaska        ') then
        do t=ibegyr,iendyr
        sup_save(t,s,ii) = 0.10
        enddo
        endif

        write(31,801) nndnme(n),nsps,ii+3,
*       (hisprc(t,s,5),t=1,(begyr-1993)),
*       (sup_save(t,s,ii),t=ibegyr,iendyr)
841     continue
```

```
851     continue

        endif

        stop
        end
```

SUBROUTINE: NUM2CHAR.FOR (NUM1,CHARVAL)

CALLED BY: INTMGN.FOR (translates LP demand, pipeline, storage and other data into MPS format to be used as input to the LP solver)
INTRPG.FOR (reads results from the solution file)

CALLS: None

READS: None

CREATES: None

MAIN THEME: The function of this routine is to convert base ten numbers to a two byte character base 62 number.

Note:

Input : decimal number (num1)

Output : base 62 two digit number (charval)
(base 62 since we are using the characters,
'0',...,'9','A',...,'Z','a',...,'z')
10 + 26 + 26 = 62
real val
integer num1,maxval,c1,c2,base
character*2 charval

Computational Remarks:
Two digit base 62 number will be of the form:
c1*(62**1) + c0*(62**0)

Step 1: Check for valid input numbers.

```
base = 62
maxval = (base-1)*base + (base-1)

if (num1 .gt. maxval) then
write(*,*)' input number to subroutine convert is too large.'
write(*,*)' subroutine halted'
stop
end if

val = (float(num1)/float(base))
```

Step 2: Compute decimal digits c1 and c2 from num1.

```
if (val .ge. 1.0) then
```

```
c1 = int(val)
c2 = num1 -c1*base
else
c1 = 0.0
c2 = num1
end if
```

Step 3: Calculate character digit for c1 and c2.

```
if   (c1 .le. 9) then     ! '0',...,'9'
charval(1:1) = char(48+c1)

else if (c1. le. 35) then ! 'A',...,'Z'
charval(1:1) = char(55+c1)

else if (c1. le. 61) then ! 'a',...,'z'
charval(1:1) = char(61+c1)

else
write(*,*)'incorrect value for c1'
stop
end if

if   (c2 .le. 9) then     ! '0',...,'9'
charval(2:2) = char(48+c2)

else if (c2. le. 35) then ! 'A',...,'Z'
charval(2:2) = char(55+c2)

else if (c2. le. 61) then ! 'a',...,'z'
charval(2:2) = char(61+c2)

else
write(*,*)'incorrect value for c2'
stop
end if

return
end
```

SUBROUTINE: PPP.FOR

CALLED BY:	INTRDT.FOR (reads in the transportation and other non-demand data for the LP)
CALLS:	ERRMSG.FOR (prints out error and warning messages)
READS:	'distind.spc' (contains the regional margins for distillate in the industrial sector) 'distele.spc' (contains the regional margins for distillate in the electical power sector) **'**refmarg.spc' (contains a summary of the refinery margins) '1pctind.spc' (contains the regional margins 1% sulfur resid in the industrial sector) '3pctind.spc' (contains the regional margins 3% sulfur resid in the industrial sector) '1pctele.spc' (contains the regional margins 1% sulfur resid in the electric power sector) '3pctele.spc' (contains the regional margins 3% sulfur resid in the electric power sector)
CREATES:	None
MAIN THEME:	This subroutine calculates petroleum product prices for the industrial and electrical power generation sectors.

Note: petroleum products

1. distillate
2. 1% sulfur resid.
3. 3% sulfur resid

demand sectors

1. industrial
2. electrical power generation

ndrg is the number of demand regions.
ntme is the number of time periods.
nname is used to read node names.
mxnval is the number of values read in from non-annualized files.
refmar is refinery margins.
regmar is regional margins.

Step 1: **Initialize margins arrays.**

```
do time=1,ntme
do product=1,3
refmar(product,time) = 0.00
do sector=1,2
do region=1,ndrg
regmar(product,sector,region,time)=0.00
end do
end do
end do
end do
```

Step 2a: **Open ‘refmarg.spc’ and read in data. ‘refmarg.spc’ contains the refinery markups from crude oil to the petroleum products for each year. Close file.**

Note: Gulf Coast refinery margins.

```
        open (unit=98,file='refmarg.spc')
        read(98,*)
        read(98,*)
        read(98,*)
        read(98,*)

        do 110 t=1,ntme
100     read(98,105) tyear,tprdis,tprone,tprthree
105     format(i4,3(5x,f5.2))
        if(tme(t).eq.tyear) then
        refmar(1,t)=tprdis
        refmar(2,t)=tprone
        refmar(3,t)=tprthree
        else if(tme(t).gt.tyear) then
        goto 100
        else
        call errmsg(4,101)
        end if

110     continue
        close(98)
```

Note: Regional margins, distillate (product=1) & industrial sector (sector=1)

Code changed to read in EXACTLY MXNVAL years of margins. Years not specified will equal the largest year not greater than the year.

Example:

Read in

1993 1995 2000 2005 2010 2015 2020
0.10 0.20 0.30 0.40 0.50 0.60 0.70

Then

1993 1994 1995 1996 1997 1998 1999 2000 2001 2002
0.10 0.10 0.20 0.20 0.20 0.20 0.20 0.30 0.30 0.30 etc.

Step 3a: **Open ‘distind.spc’. ‘distind.spc’ contains the regional margins for distillate (product = 1) in the industrial sector (sector = 1). Read in data. Close file.**

```
          open(unit=98,file='distind.spc')

          read(98,900) dummy1
          read(98,905) (tyrs(time),time=1,mxnval)
260       read(98,920,end=280) nname,(temp(time),time=1,mxnval)
          do 270, region=1,ndrg
          n=dmnpnt(region)
          if(nname.ne.nndnme(n)) go to 270
          tyr=1
          do time=1,ntme
265       tyr=tyr
          if((tyr.eq.mxnval).or.(tme(time).lt.tyrs(tyr+1))) then
          regmar(1,1,region,time)=temp(tyr)
          else
          tyr=tyr+1
          goto 265
          endif
          end do
          go to 260
270       continue
          write(*,930) nname



          call errmsg(3,204)
          go to 260

280       close(98)
```

Note: Code changed to read in EXACTLY MXNVAL years of margins. Years not specified will equal the largest year not greater than the year.

Example:

Read in

1993 1995 2000 2005 2010 2015 2020
0.10 0.20 0.30 0.40 0.50 0.60 0.70

Then

1993 1994 1995 1996 1997 1998 1999 2000 2001 2002
0.10 0.10 0.20 0.20 0.20 0.20 0.20 0.30 0.30 0.30 etc.

Step 3b: **Open ‘1pctind.spc’. ‘1pctind.spc’ contains the regional margins for 1% sulfur resid (product=2) in the industrial sector (sector=1). Read in data. Close file.**

```
          open(unit=98,file='1pctind.spc')
```

```
        read(98,900) dummy1
        read(98,905) (tyrs(time),time=1,mxnval)
360     read(98,920,end=380) nname,(temp(time),time=1,mxnval)
        do 370,region=1,ndrg
        n=dmnpnt(region)
        if(nname.ne.nndnme(n)) go to 370
        tyr=1
        do time=1,ntme
365     tyr=tyr
        if((tyr.eq.mxnval).or.(tme(time).lt.tyrs(tyr+1))) then
        regmar(2,1,region,time)=temp(tyr)
        else
        tyr=tyr+1
        goto 365
        endif
        end do
        go to 360
        continue
        write(*,930) nname

        call errmsg(3,204)
        go to 360

380     close(98)
```

Note: Code changed to read in EXACTLY MXNVAL years of margins. Years not specified will equal the largest year not greater than the year.

Example:

Read in

1993 1995 2000 2005 2010 2015 2020
0.10 0.20 0.30 0.40 0.50 0.60 0.70

Then

1993 1994 1995 1996 1997 1998 1999 2000 2001 2002
0.10 0.10 0.20 0.20 0.20 0.20 0.20 0.30 0.30 0.30 etc.

Step 3c: Open ‘3pctind.spc’. ‘3pctind.spc’ contains the regional margins for 3% sulfur resid (product=3) in the industrial sector (sector=1). Read in data. Close file.

```
        open(unit=98,file='3pctind.spc')

        read(98,900) dummy1
        read(98,905) (tyrs(time),time=1,mxnval)
460     read(98,920,end=480) nname,(temp(time),time=1,mxnval)
        do 470,region=1,ndrg
        n=dmnpnt(region)
        if(nname.ne.nndnme(n)) go to 470
        tyr=1
        do time=1,ntme
465     tyr=tyr
```

```
        if((tyr.eq.mxnval).or.(tme(time).lt.tyrs(tyr+1))) then
        regmar(3,1,region,time)=temp(tyr)
        else
        tyr=tyr+1
        goto 465
        endif
        end do
        go to 460
470     continue
        write(*,930) nname

        call errmsg(3,204)
        go to 460

480     close(98)
```

Note: Code changed to read in EXACTLY MXNVAL years of margins. Years not specified will equal the largest year not greater than the year.

Example:

```
Read in
     1993 1995 2000 2005 2010 2015 2020
     0.10 0.20 0.30 0.40 0.50 0.60 0.70
Then
     1993 1994 1995 1996 1997 1998 1999 2000 2001 2002
     0.10 0.10 0.20 0.20 0.20 0.20 0.20 0.30 0.30 0.30 etc.
```

Step 4a: Open ‘distele.spc’ and read in data. ‘distele.spc’ contains the regional margins for distillate (product=1) in the electric power generation sector (sector=2). Close file.

```
        open(unit=98,file='distele.spc')

        read(98,900) dummy1
        read(98,905) (tyrs(time),time=1,mxnval)
560     read(98,920,end=580) nname,(temp(time),time=1,mxnval)
        do 570,region=1,ndrg
        n=dmnpnt(region)
        if(nname.ne.nndnme(n)) go to 570
        tyr=1
        do time=1,ntme
565     tyr=tyr
        if((tyr.eq.mxnval).or.(tme(time).lt.tyrs(tyr+1))) then
        regmar(1,2,region,time)=temp(tyr)
        else
        tyr=tyr+1
        goto 565
        endif
        end do
        go to 560
570     continue
        write(*,930) nname

        call errmsg(3,204)
        go to 560

580     close(98)
```

Note: Code changed to read in EXACTLY MXNVAL years of margins. Years not specified will equal the largest year not greater than the year.

Example:

```
    Read in
        1993 1995 2000 2005 2010 2015 2020
        0.10 0.20 0.30 0.40 0.50 0.60 0.70
    Then
        1993 1994 1995 1996 1997 1998 1999 2000 2001 2002
        0.10 0.10 0.20 0.20 0.20 0.20 0.20 0.30 0.30 0.30 etc.
```

Step 4b: **Open '1pctele.spc' and read in data. '1pctele.spc' contains the regional margins for 1% sulfur resid (product=2) in the electric power generation sector (sector=2). Close file.**

```
        open(unit=98,file='1pctele.spc')

        read(98,900) dummy1
        read(98,905) (tyrs(time),time=1,mxnval)
660     read(98,920,end=680) nname,(temp(time),time=1,mxnval)
        do 670,region=1,ndrg
        n=dmnpnt(region)
        if(nname.ne.nndnme(n)) go to 670
        tyr=1
        do time=1,ntme
665     tyr=tyr
        if((tyr.eq.mxnval).or.(tme(time).lt.tyrs(tyr+1))) then
        regmar(2,2,region,time)=temp(tyr)
        else
        tyr=tyr+1
        goto 665
        endif
        end do
        go to 660
670     continue
        write(*,930) nname

        call errmsg(3,204)
        go to 660

680     close(98)
```

Note: Code changed to read in EXACTLY MXNVAL years of margins. Years not specified will equal the largest year not greater than the year.

Example:

Read in

```
1993 1995 2000 2005 2010 2015 2020
0.10 0.20 0.30 0.40 0.50 0.60 0.70
```
Then
```
1993 1994 1995 1996 1997 1998 1999 2000 2001 2002
0.10 0.10 0.20 0.20 0.20 0.20 0.20 0.30 0.30 0.30 etc.
```

Step 4c: **Open ‘3pctele.spc’ and read in data. ‘3pctele.spc’ contains regional margins for 3% sulfur resid (product=3) in the electric power generation sector (sector=2). Close file.**

```
        open(unit=98,file='3pctele.spc')

        read(98,900) dummy1
        read(98,905) (tyrs(time),time=1,mxnval)
760     read(98,920,end=780) nname,(temp(time),time=1,mxnval)
        do 770,region=1,ndrg
        n=dmnpnt(region)
        if(nname.ne.nndnme(n)) go to 770
        tyr=1
        do time=1,ntme
765     tyr=tyr
        if((tyr.eq.mxnval).or.(tme(time).lt.tyrs(tyr+1))) then
        regmar(3,2,region,time)=temp(tyr)
        else
        tyr=tyr+1
        goto 765
        endif
        end do
        go to 760
770     continue
        write(*,930) nname

        call errmsg(3,204)
        go to 760

780     close(98)

900     format(a80)
905     format(t20,7(2x,i4))
920     format(a20,t21,7(f5.2,1x))
930     format(' Demand region: ',a20)

        return
        end
```

SUBROUTINE: PPRICE2.FOR

CALLED BY: INTMGN.FOR (translates LP demand, pipeline, storage and other data into MPS format to be used as input to the LP solver)
INTRPS.FOR (produces summary supply and demand reports)

CALLS: None

READS: None

CREATES: None

MAIN THEME: This subroutine is used in the calculation of petroleum product prices.

Note:

petroleum products:
1. distillate
2. 1% sulfur resid.
3. 3% sulfur resid

demand sectors:
1. industrial
2. electrical power generation

pcrude is the price of crude in $/barrel.
pprice is petroleum product price in $/MCF.

Inputs to program:

1. Gulf Coast price of crude ($/barrel)

2. Refinery margins (specific to a petroleum product)

3. Regional margins (specific to a petroleum product, demand sector, region)

Outputs from program:

1. End-use price (specific to a petroleum product, demand sector, region)

Calculations:

conversion factor

1 barrel = 5.8 MMBTU, 1 MMTBU = 1/(1.03)MCF so
1 barrel = 5.8/1.03 MCF

Step 1: **Convert price of crude in $/BBL to $/MCF.**

```
pcrude = (pcrude/5.8)*1.03
```

Step 2: **Calculate appropriate end-use price. End subroutine.**

Note: indmarg = 0.0 if calculations are for elec. power sector.

```
ppprice = (pcrude/5.8)*1.03+refmarg+regmarg+indmarg
return
end
```

SUBROUTINE: RGET.FOR (CNAME, RVAL, RDUAL, C1, C2, C3, C4, C5, C6, CODE, MXC)

CALLED BY: INTRPG.FOR (reads results from solution file)

CALLS: None

READS: ‘gasall.prt’ (LP solution file)

CREATES: None

MAIN THEME: Reads the records from the LP solution file.

Note: rname is the row name.
rval is the optimal row value.
rdual is the optimal dual (LaGrange multiplier) value that is passed into the routine.
cname is the constraint name.

Example: cname = 'MB###   ' (material balance constraint)
rname = 'MB311   ' (material balance constraint)

Step 1: **Read from LP solution file (in MPS format). Check if it is a valid row name, read the rest of the row.**

```
1         read(14,11) rname
11        format(t4,a8)
          if ((rname.eq.'Number  ').or.(rname.eq.'        ').or.
          * (rname.eq.'umns Sec')) goto 1

          backspace(14)

          read(14,12) rname,rval,rdual
12        format(t12,a8,t26,f12.6,t54,f13.6)
```

Note: A maximum of 8 characters for an MPS variable name.

```
          j=0
```

Step 2: **Check for name mismatch.**

```
          do 100 i=1,8
          if(rnme1(i).eq.cname(i)) go to 100
          if(cname(i).eq.spch) go to 50
10        write(*,13) rname,cname
13        format(' *** mismatch on row name: ',a8,1x,8a1)
          write(*,*)'i       =',i
          write(*,*)'j       =',j
          write(*,*)'rname    =',rname
```

```
        write(*,*)'rnme1    =',rnme1
        write(*,*)'cname    =',cname
        write(*,*)'c1       =',c1
        write(*,*)'c2       =',c2
        write(*,*)'c3       =',c3
        write(*,*)'c4       =',c4
        write(*,*)'c5       =',c5
        write(*,*)'c6       =',c6
        write(*,*)'code(c1) =',code(c1)
        write(*,*)'code(c2) =',code(c2)
        write(*,*)'code(c3) =',code(c3)
        write(*,*)'code(c4) =',code(c4)
        write(*,*)'code(c5) =',code(c5)
        write(*,*)'code(c6) =',code(c6)
        stop
50      j=j+1
        if(j.gt.6) stop ' too many indices in row name'
        if((j.eq.1).and.(rnme1(i).ne.code(c1))) go to 10
        if((j.eq.2).and.(rnme1(i).ne.code(c2))) go to 10
        if((j.eq.3).and.(rnme1(i).ne.code(c3))) go to 10
        if((j.eq.4).and.(rnme1(i).ne.code(c4))) go to 10
        if((j.eq.5).and.(rnme1(i).ne.code(c5))) go to 10
        if((j.eq.6).and.(rnme1(i).ne.code(c6))) go to 10
100     continue

        return
        end
```

SUBROUTINE: SROM.FOR (STORAGE RESERVOIR OPERATOR MODULE (SROM))

CALLED BY:	INTRDT.FOR (reads in the transportation and other non-demand data for the LP)
CALLS:	ERRMSG.FOR (prints out error and warning messages) GET_OPT.FOR (calculates the costs and revenues of the storage extraction profiles, and determines which one has the highest net profit) REGNUM (calculates the demand region numbers from the storage reservoir id)
READS:	'srom.in' (specifies the properties of individual storage reservoirs, such as levelized cost, fixed O & M cost, variable O & M cost, maxinum extraction rate for the season i as a percentage of the working gas per day by option and region, and the maxinum injection rate for season four as a percentage of working gas per day by option and region) 'stor.log' (writes out data that was read in or computed as a check) 'storvals.spc' (contains the percentage of storage capacity to be used by the model, and the decline percentage for the extraction rate from the storage reservoirs)
CREATES:	None
MAIN THEME:	This subroutine reads in storage reservoir-level information and determines which of three storage options is most economically interesting to the storage reservoir operator.
Note:	n is the node number. v is the reservoir number. nsto(n,t) is the number of storage reservoirs available at node n,time t. resvfyr is the first year that reservoir storage is available. resvid is the reservoir identification field. mvcp is the maximum workable volume capacity of storage increment. mecp is the maximum extraction capacity (% of storage capacity). micp is the maximum injection capacity (% of storage capacity). lcst is thelevelized investment costs for capacity. fom is the fixed O & M costs for capacity vom is the variable O & M costs for injection/extraction combined. fus is the fuel used for injection/extraction to/from storage (%). decline is the injection/extraction decline rate per year.

struse is the % of storage capacity to be used by the model.
ntme is the number of time periods.
strfyr is the first year that a reservoir is available.
resvid is the reservoir identification field.

Step 1a: **Open 'storvals.spc'. Read in data. This file contains the percentage of storage capacity to be used by the model, and the decline percentage for the extraction rate from the storage reservoirs. Close file.**

```
        open(unit=88,file='storvals.spc')
        read(88,*)decline
        read(88,*)storuse
        decline = decline/100.
        storuse = storuse/100.

        close(88)
```

Step 2a: **Open reservoir-level input file 'srom.in'. 'srom.in' specifies the properties of individual storage reservoirs for each option type.**

```
        nin = 88
        open(nin,file='srom.in')
```

Step 3a: **Open log file. 'stor.log' writes out storage data that was read in or computed as a check.**

```
        nlog =87
        open(nlog,file='stor.log')
```

Step 3b: **Initialize number of storage reservoirs for each (year, node) combination.**

```
        do 410 n=1,nnde
        do 405 t=1,ntme
        nsto(n,t)=0
        do v=1,mxnres
        strfyr(n,t,v) = 0
        enddo  ! v loop
405     continue
410     continue
```

Step 3c: **Read headers.**

```
        do i=1,11
        read(nin,700) dummy
        end do

700     format(a165)
```

Step 3d: Read reservoir-level data.

Note: Existing reservoirs must preceed new ones for the given data structures.

```
20      read(nin,'(a11)',end=60) resvid
        backspace(nin)
        read(nin,*)dummy11,resvfyr,mvcp,ratio,nyr,fus,
*       lc(1),fom(1),vom(1),mer1(1),mer2(1),mer3(1),mir(1),
*       lc(2),fom(2),vom(2),mer2(2),mer3(2),mir(2),
*       lc(3),fom(3),vom(3),mer1(3),mir(3)

        if (vom(1).le.0.0) vom(1) = machep
        if (vom(2).le.0.0) vom(2) = machep
        if (vom(2).le.0.0) vom(3) = machep
```

Note: Option 1 all data as read in.

Note: Option 2.

```
        mer1(2) = mer2(2)  ! seasons 1 and 2 have the same extraction rate
```

Note: Option 3.

```
        mer2(3) = mer1(3)
        mer3(3) = mer1(3)  ! seasons 1, 2, and 3 have the same extraction rate
```

Step 3e: Compute the region number from the reservoir id and compute the extraction rates.

```
        call regnum(resvid,n)
```

Note: Option 1.

```
        extract(1) = mer1(1)   ! option 1, season 1   (5 days)
        extract(2) = mer2(1)   ! option 1, season 2   (26 days)
        extract(3) = mer3(1)   ! option 1, season 3   (90 days)
```

Note: Option 2.

```
        extract(4) = mer1(2)   ! option 2, seasons 1 & 2 (31 days)
        extract(5) = mer3(2)   ! option 2, season  3
```

Note: Option 3.

```
        extract(6) = mer1(3)   ! option 3, seasons 1,2, & 3 (121 days)
```

Step 3f: Check the validity of the extraction rates.

```
        do i=1,6
        if (extract(i).lt.0.0) then
        write(*,*)'extract(',i,')=',extract(i)
        write(*,*)'resvid=',resvid
        stop
        endif
        enddo
```

Step 3g: **Assign seasonal prices and compute which storage option is most economically attractive for each reservoir. Use dual prices for region (n), time (t), and season (l) (output of LP).**

```
        do t=1,ntme
        price(1) = duals(n,t,1)
        price(2) = duals(n,t,2)
        price(3) = duals(n,t,3)
```

Note: Decide to use the first year's decision on storage option for the remaining years.

```
        if (t.eq.ibegyr) then
        call get_opt(price,extract,mvcp,lc,vom,opt,profit)
        endif

        option(t) = opt

        write(nlog,21)n,t,resvid,resvfyr,price(1),price(2),price(3),
*       extract(1),extract(2),extract(3),extract(4),extract(5),
*       extract(6),
*       lc(1),lc(2),lc(3),vom(1),vom(2),vom(3),mvcp,option(t),
*       profit(1),profit(2),profit(3)
21      format(I2,1x,I2,a11,1x,I4,15(f6.2,1x),f10.2,1x,i2,
*       1x,3(f12.2,1x))
        end do
```

Step 3h: **Calculate the effective maximum storage capacity.**

```
        mvcp = storuse*mvcp
```

Step 3i: **Calculate the first GSAM year that the reservoir is available.**

```
40      do t=1,ntme
        if (resvfyr.le.tme(t)) then

        do t1 = t,ntme
        nsto(n,t1)  = nsto(n,t1) + 1
        if (nsto(n,t1) .gt. mxnres) then
        write(*,*)'maximum # of reservoirs exceeded'
        write(*,*)'nsto(',n,',',t1,')=',nsto(n,t1)
        write(*,*)'maximum # of reservoirs=',mxnres
        call errmsg(4,801)
        end if ! nsto(n,t1) .gt. mxnres
        end do   ! t1 loop
        go to 55
        end if
        end do       ! t loop
```

Note: The reservoir first year is beyond the last GSAM year, therefore the reservoir should not be counted.

```
        write(nlog,*)'resvfyr=',resvfyr
        write(nlog,*)'last GSAM year=', tme(ntme)
        write(nlog,*)'reservoir ',resvid,' has been excluded'
        go to 20
```

Step 3j: Store values in arrays.

```
55      do t1=t,ntme
        nres              = nsto(n,t1)
        strind(n,t1,nres)  = resvid
        strvcp(n,t1,nres)  = mvcp
        strfyr(n,t1,nres)  = t
```

Step 3k: Compute storage extraction rates for each extraction season 1,2,3; otherwise set equal to 0.

```
        opt = option(t1)  ! select the appropriate option

        strecp(n,t1,nres,1)= mer1(opt)*((1.0-decline)**(tme(t1)-tme(t)))
        strecp(n,t1,nres,2)= mer2(opt)*((1.0-decline)**(tme(t1)-tme(t)))
        strecp(n,t1,nres,3)= mer3(opt)*((1.0-decline)**(tme(t1)-tme(t)))
        strecp(n,t1,nres,4)= 0.0
```

Step 3l: Compute Storage injection rate for injection season 4; otherwise set equal to 0.

```
        stricp(n,t1,nres,1)= 0.0
        stricp(n,t1,nres,2)= 0.0
        stricp(n,t1,nres,3)= 0.0
        stricp(n,t1,nres,4)= mir(opt)*((1.0-decline)**(tme(t1)-tme(t)))
```

Step 3m: Add the cost of the base gas to the levelized cost for new reservoirs only.

Note: The cost is for the first year that the reservoir is available.
ratio is the base gas /present value of working gas.
nyr is the number of years for the life of the reservoir.
duals(n,t,l) is the dual prices to material balance constraints ($/MCF).
duals(n,t,4) is off-season price (season 4) for gas.

```
        strcst(n,t1,nres)  = lc(opt) + duals(n,t,4)*ratio
        strfom(n,t1,nres)  = fom(opt)
        strvom(n,t1,nres)  = vom(opt)
        strfus(n,t1,nres)  = fus
        end do
        go to 20
```

Step 3n: Write to storage log file.

```
60      do n=1,nnde
```

```
do t=1,ntme
if  (nsto(n,t).gt.0)  then
write(nlog,*)'nsto(',n,',',t,')=',nsto(n,t)
do v=1,nsto(n,t)
write(nlog,*)'n,t,v,strvcp',n,t,v,strvcp(n,t,v)
end do
end if
enddo
enddo

close(nout)
close(nlog)
close(nin)
close(nefh)
return
end
```

SUBROUTINE: REGNUM (RESVID,N)

CALLED BY:	SROM.FOR (reads in storage reservoir-level information and determines which of the three storage options is most economically advantageous to the storage reservoir operator)
CALLS:	None
READS:	None
CREATES:	None
MAIN THEME:	Calculate the demand region numbers from the storage reservoir id.

```
character*11 resvid
integer n


if (resvid(1:2).eq.'01') then
n = 1
return
else if (resvid(1:2).eq.'02') then
n = 2
return
else if (resvid(1:2).eq.'03') then
n = 3
return
else if (resvid(1:2).eq.'04') then
n = 4
return
else if (resvid(1:2).eq.'05') then
n = 5
return
else if (resvid(1:2).eq.'06') then
n = 6
return
else if (resvid(1:2).eq.'07') then
n = 7
return
else if (resvid(1:2).eq.'08') then
n = 8
return
else if (resvid(1:2).eq.'09') then
n = 9
return
else if (resvid(1:2).eq.'10') then
n =10
return
else if (resvid(1:2).eq.'11') then
n =11
return
else if (resvid(1:2).eq.'12') then
n = 12
return
```

```
else if (resvid(1:2).eq.'13') then
n = 13
return
else if (resvid(1:2).eq.'14') then
n = 14
return
else if (resvid(1:2).eq.'15') then
n = 15
return
else
write(*,*)'region mismatch, resvid=',resvid
stop
end if
end
```

SUBROUTINE: SUP_REP.FOR

CALLED BY: INTRPT.FOR (produces output files)

CALLS: None

READS: None

CREATES: **‘**gsamsln.sup’ (contains a summary of storage use and peaking supply information such as LNG and propane)

MAIN THEME: Produce Reports for peak supply (propane/air and LNG), storage, and pipeline usage.

Note: ndrg is the number of demand regions.
nlds is the number of load segments.
decline is the extraction/injection decline rate per year.
storuse is the percentage of storage capacity to be used in the model.
nndme is the identifier of nodes.
npks is the number of peak supply options.
mxnres is the maxinum number of storage reservoirs for a fixed node and year.
ndsex is the storage extraction summed over t years.
pklc is the levelized investment cost.
strvcp is the maxinum volume capacity of storage.
strecp is the maxinum extraction capactiy as a percentage of storage capacity for the load season.
nlnk is the number of transport links.
tnfvl is the transportation forward flow read from the LP.
ldsdy is the number of days in the gas load season.

Step 1: **Set positivity tolerance**

```
tol = 0.0
```

Step 2a: **Open output file. ‘gsamsln.sup’ contains a summary of storage use and peaking supply information such as LNG and propane.**

```
open(unit=89,file='gsamsln.sup')

forward  ='>>>>>>>>>>>>>>>>>>>>'
backward ='<<<<<<<<<<<<<<<<<<<<'
```

Step 2b: Initialize pipeline values.

```
        do n=1,ndrg
```

Note: l is the gas load segment index (l=1,nlds)=> MMCF/day data (l=nlds+1)=> BCF/year data

```
        do l=1,nlds+1
        counter = 0

        do t=ibegyr,iendyr
        flowin(t)  = 0.0
        flowout(t) = 0.0
        cap(t)     = 0.0
        enddo  ! t loop
```

Step 2c: Write out headers.

```
        counter = counter +1
        write(89,*)' ',counter
        write(89,90)decline*100.0,storuse*100.0
90      format(' Decline factor: ',1x,f5.1,'%'1x,
*       ' Storage Usage: ',f5.1,'%')
        write(89,100) nndnme(n)
100     format(' Demand Region : ',a20)

        if(l.le.nlds) then
        counter = counter + 1
        write(89,*)' ',counter
        write(89,105) forward,l,ldsdy(l),backward
105     format(/a20,' Load Period: ',i2,
*       ' Number of Days:',i3,' (MMCF/day)',a20)
        else
        counter = counter + 1
        write(89,*)' ',counter
        title='Annual Averages (BCF)'
        write(89,110)forward,title,backward
110     format(/a20,a25,a20)
        endif

        write(89,115) (tme(t),t=ibegyr,iendyr)
115     format(' ',t42,33(3x,i4,2x))
```

Step 2d: Initialize usage variables and calculate peak supply.

```
        do p=1,npks
        do t=ibegyr,iendyr
        ndsex(t)    = 0.0
        ndsin(t)    = 0.0
        do v=1,mxnres
        totstor(v,t) = 0.0
        end do  ! v loop
        ndpks(t)= 0.0
        do k=1,2
        pk_prc(p,t,k) = 0.0
        sum(k) = sum(k)
        enddo !k loop
        enddo   ! t loop
        enddo     ! p loop


        ifl=0
        do p=1,npks
        do t=ibegyr,iendyr
        if(l.le.nlds) then     ! in MMCF/day
        olduse(t)      = pkovl(p,n,t,l,1)
        if (pkovl(p,n,t,l,1).gt.tol) then  ! only compute values if actual usage
        olddel(t)     = pkd(p,n,1)
        pk_prc(p,t,1) = (pklc(p,n,1)/365.0)+pkvc(p,n,1)+
*       pkfc(p,n,1)
        else
        olddel(t)     = pkd(p,n,1)
        pk_prc(p,t,1) = 99999.99
        endif


        newuse(t)     = pkovl(p,n,t,l,2)
        if (pkovl(p,n,t,l,2).gt.tol) then  ! only compute values if actual usage
        newdel(t)     = pkd(p,n,2)
        pk_prc(p,t,2) = (pklc(p,n,2)/365.0)+pkvc(p,n,2)+
*       pkfc(p,n,2)
        else
        newdel(t)     = pkd(p,n,2)
        pk_prc(p,t,2) = 99999.99
        endif

        totpeak(t)= olduse(t)+newuse(t)    ! existing + new

        else                    ! in BCF/year
        totpeak(t)=0.0
        olduse(t) =0.0
        newuse(t) =0.0
        olddel(t) =0.0
        newdel(t) =0.0
        do l1=1,nlds
        olduse(t) = olduse(t) +
*       pkovl(p,n,t,l1,1)*ldsdy(l1)/1000.
        newuse(t) = newuse(t) +
*       pkovl(p,n,t,l1,2)*ldsdy(l1)/1000.

        olddel(t) = olddel(t) +
*       pkd(p,n,1)*ldsdy(l1)/1000.
        newdel(t) = newdel(t) +
*       pkd(p,n,2)*ldsdy(l1)/1000.

        enddo  !l1 loop
        totpeak(t)  = totpeak(t)+ olduse(t) + newuse(t)
        if (olduse(t).gt.tol) then

        pk_prc(p,t,1) = (pklc(p,n,1)/365.0)+pkvc(p,n,1)+
*       pkfc(p,n,1)
        else
        pk_prc(p,t,1)  = 99999.99
        endif

        if(newuse(t).gt.tol) then
        pk_prc(p,t,2) = (pklc(p,n,2)/365.0)+pkvc(p,n,2)+
*       pkfc(p,n,2)
```

```
        else
        pk_prc(p,t,2) = 99999.99
        endif



        endif ! l.le.nlds

        ndpks(t)=ndpks(t)+totpeak(t)
        enddo  ! t loop

        if(ifl.eq.0) then
        write(89,120)
120     format(/' Peaking Supply Sources:')
        ifl=1
        endif

        counter = counter+1
        write(89,*)' ',counter
        if (p.eq.1) then
        write(89,*)'Propane:'
        else
        write(89,*)'LNG:'
        end if
        counter = counter+1
        write(89,*)' ',counter

        write(89,125) (olduse(t),t=ibegyr,iendyr)
        write(89,130) (olddel(t),t=ibegyr,iendyr)
        write(89,132) (pk_prc(p,t,1),t=ibegyr,iendyr)
        counter = counter +1
        write(89,*)' ',counter
        write(89,135) (newuse(t),t=ibegyr,iendyr)
        write(89,140) (newdel(t),t=ibegyr,iendyr)
        write(89,142) (pk_prc(p,t,2),t=ibegyr,iendyr)
        counter = counter+1
        write(89,*)' ',counter
        write(89,145) (totpeak(t),t=ibegyr,iendyr)
        write(89,150) (olddel(t)+newdel(t),t=ibegyr,iendyr)
        enddo      ! p loop

        if(ifl.eq.1) then
        write(89,155) (ndpks(t),t=ibegyr,iendyr)
        endif

125     format(' ','Existing  ',1x,'Usage      ',t42,33(1x,f8.2))
130     format(' ','Existing  ',1x,'Maximum ',   t42,33(1x,f8.2))
132     format(' ','Existing  ',1x,'Price ($/MCF)',t42,33(1x,f8.2))
135     format(' ','New       ',1x,'Usage      ',t42,33(1x,f8.2))
140     format(' ','New       ',1x,'Maximum ',   t42,33(1x,f8.2))
142     format(' ','New       ',1x,'Price ($/MCF)',t42,33(1x,f8.2))
145     format(' ','Total     ',1x,'Usage      ',t42,33(1x,f8.2))
150     format(' ','Total     ',1x,'Maximum ',   t42,33(1x,f8.2))
155     format(/' Total Peaking Supply Usage: ',t42,33(1x,f8.2))
```

Note: sevl is the storage extraction value, read from the LP (MMCF/day).
storext is the storage extraction activity in MCF.
totext is the total storage extraction for the year in MCF.
strcst is the levelized investment cost in $/MCF.

strfom is the fixed O&M cost in $/MCF.
strvom is the variable O&M in $/MCF.
fuscst is the fuel useage/shrinkage cost in $.

Step 2e: Calculate the maximum number of reservoirs for node for all times t.

```
nres = 0
do t=ibegyr,iendyr
if (nsto(n,t).gt.nres) nres = nsto(n,t)

enddo ! t loop
```

Step 2f: Initialize and calculate storage extraction.

```
do t=ibegyr,iendyr
storcap(t) = 0.0
avgcost(t) = 0.0
totwt      = 0.0
if(nsto(n,t).gt.0) then
do v = 1,nsto(n,t)
levcst     = 0.0
fomcst     = 0.0
vomcst     = 0.0
totext     = 0.0
do l1=1,nlds
storext = sevl(n,t,v,l1)*ldsdy(l1)*1000.0
totext  = totext + storext
totwt   = totwt+ storext
levcst  = levcst + strcst(n,t,v)*storext
fomcst  = fomcst + strfom(n,t,v)*storext
vomcst  = vomcst + strvom(n,t,v)*storext

enddo  ! l1 loop
fuscst=duals(n,t,4)*totext*strfus(n,t,v)/100.0
999     format(1x,i4,1x,a20,1x,4(f10.1,1x))
avgcost(t) = avgcost(t) +
*       (levcst+fomcst+vomcst+fuscst)
```

Step 2g: Accumulate storage extraction values to be printed out.

```
if(l.le.nlds) then        ! in MMCF/day
totstor(v,t)= sevl(n,t,v,l)
if (totstor(v,t).gt.tol) then
vcp(v,t)  = strvcp(n,t,v)*strecp(n,t,v,l)/100.0
else
vcp(v,t)  = 0.0
endif
else
totstor(v,t)= 0.0         ! in BCF/year
vcp(v,t)    = 0.0
sumv        = 0.0
do  l1=1,nlds
sumv = sumv + sevl(n,t,v,l1)
enddo

do  l1=1,nlds
```

```
        totstor(v,t)=totstor(v,t)+
*       sevl(n,t,v,l1)*ldsdy(l1)/1000.0
        if (sumv.gt.tol) then
        vcp(v,t) =vcp(v,t)+
*       (strvcp(n,t,v)*(strecp(n,t,v,l1)/100.0)*
*       ldsdy(l1))/1000.0

        endif
        enddo ! l1 loop
```

Note: Can not exceed the volume capacity and can't exceed the annual (weighted) daily extraction rate.

```
        vcp(v,t) = min(vcp(v,t),strvcp(n,t,v)/1000.)

        endif
        ndsex(t)=ndsex(t)+totstor(v,t)
        storcap(t) = storcap(t) + vcp(v,t)
        enddo    ! v loop

        endif

        if (totwt.gt.tol) then
        avgcost(t) = avgcost(t)/totwt
        else
        avgcost(t) = 0.0
        end if
        enddo  ! t loop

        counter = counter+1
        write(89,*)' ',counter
        write(89,160) (ndsex(t),t=ibegyr,iendyr)

160     format(' Total Storage Extraction: ',t42,33(1x,f8.2))

        write(89,170)(storcap(t),t=ibegyr,iendyr)
170     format(' Maximum Possible: ',t42,33(1x,f8.2))

        write(89,200)(avgcost(t),t=ibegyr,iendyr)
200     format (' ','Avg. Price ($/MCF)',t42,33(1x,f8.2))
```

Step 2h: Pipeline calculations.

```
        do q=1,nlnk
        o=lnkpnt(1,q)
        d=lnkpnt(2,q)
        if (d.eq.n) then
        nname=nndnme(o)
        elseif (o.eq.n) then
        nname=nndnme(d)
        endif
        v0=lnkfus(q)/100.0

        if ((o.eq.n).or.(d.eq.n)) then
        do t=ibegyr,iendyr
        if(d.eq.n) then
        if(l.le.nlds) then
        flowin(t) =flowin(t) + (1.0-v0)*tnfvl(q,t,l)
        flowout(t)=flowout(t)+ tnrvl(q,t,l)
        cap(t)    =cap(t)    + trncap(q,t)
        else
        sumin  = 0.0
        sumout = 0.0
        sumcap = 0.
        do l1=1,nlds
        v1= ldsdy(l1)/1000.
        sumin =sumin  + (1.0-v0)*tnfvl(q,t,l1)*v1
        sumout=sumout + tnrvl(q,t,l1)*v1
        sumcap=sumcap + trncap(q,t)*v1
        enddo  ! l1 loop
        flowin(t)  = flowin(t)  + sumin
        flowout(t) = flowout(t) + sumout
        cap(t)= cap(t)     + sumcap
        endif    ! l.le.nlds
        else
        if(l.le.nlds) then
        flowin(t) =flowin(t) + (1.0-v0)*tnrvl(q,t,l)
        flowout(t)=flowout(t)+ tnfvl(q,t,l)
        cap(t)    =cap(t)    + trncap(q,t)
        else
        sumin  = 0.0
        sumout = 0.0
        sumcap = 0.0
        do l1=1,nlds
        v1= ldsdy(l1)/1000.
        sumin =sumin  + (1.0-v0)*tnrvl(q,t,l1)*v1
        sumout=sumout + tnfvl(q,t,l1)*v1
        sumcap=sumcap + trncap(q,t)*v1
        enddo   ! l1 loop
        flowin(t) = flowin(t) + sumin
        flowout(t)= flowout(t)+ sumout
        cap(t)= cap(t)    + sumcap
        endif     ! l.le.nlds
        endif       ! d.eq.n
        enddo         ! t loop
        endif           ! (o.eq.n).or.(d.eq.n)
        enddo             ! q loop

        counter = counter+1
        write(89,*)' ',counter
        write(89,300) (flowin(t),t=ibegyr,iendyr)
300     format(' Total Pipeline Flows In',t42,33(1x,f8.2))

        write(89,305) (flowout(t),t=ibegyr,iendyr)
305     format(' Total Pipeline Flows Out',t42,33(1x,f8.2))

        write(89,310) (flowin(t)-flowout(t),t=ibegyr,iendyr)
310     format(' Net Pipeline Flows In',t42,33(1x,f8.2))

        write(89,320)(cap(t),t=ibegyr,iendyr)
320     format(' Maximum Capacity',t42,33(1x,f8.2))


        do ii=1,23
        write(89,*)' ',counter+ii
        enddo
        enddo  ! l loop
        enddo    ! n loop


        close(89)
        return
        end
```

SUBROUTINE: ZAP.FOR (ARR,NI,NJ,NK,NL,NM)

CALLED BY: INTRDT.FOR (reads in the transportation and other non-demand data for the LP)
INTRPG.FOR (reads results for the solution file)

CALLS: None

READS: None

CREATES: None

MAIN THEME: Subroutine to zero out an array.

Step 1: **Zero out array. End subroutine.**

```
        integer ni,nj,nk,nl,nm,i,j,k,l,m
        real arr(ni,nj,nk,nl,nm)

        do 110 m=1,nm
        do 100 l=1,nl
        do 90 k=1,nk
        do 80 j=1,nj
        do 70 i=1,ni
        arr(i,j,k,l,m)=0.0
70      continue
80      continue
90      continue
100     continue
110     continue

        return
        end
```